The Storied Past of QuickTime's Programming

MacTech Magazine
February • 2006

MACTECH®

The Journal of Macintosh Technology

Not CurSED, BlesSED: The keys to sed can be disguised

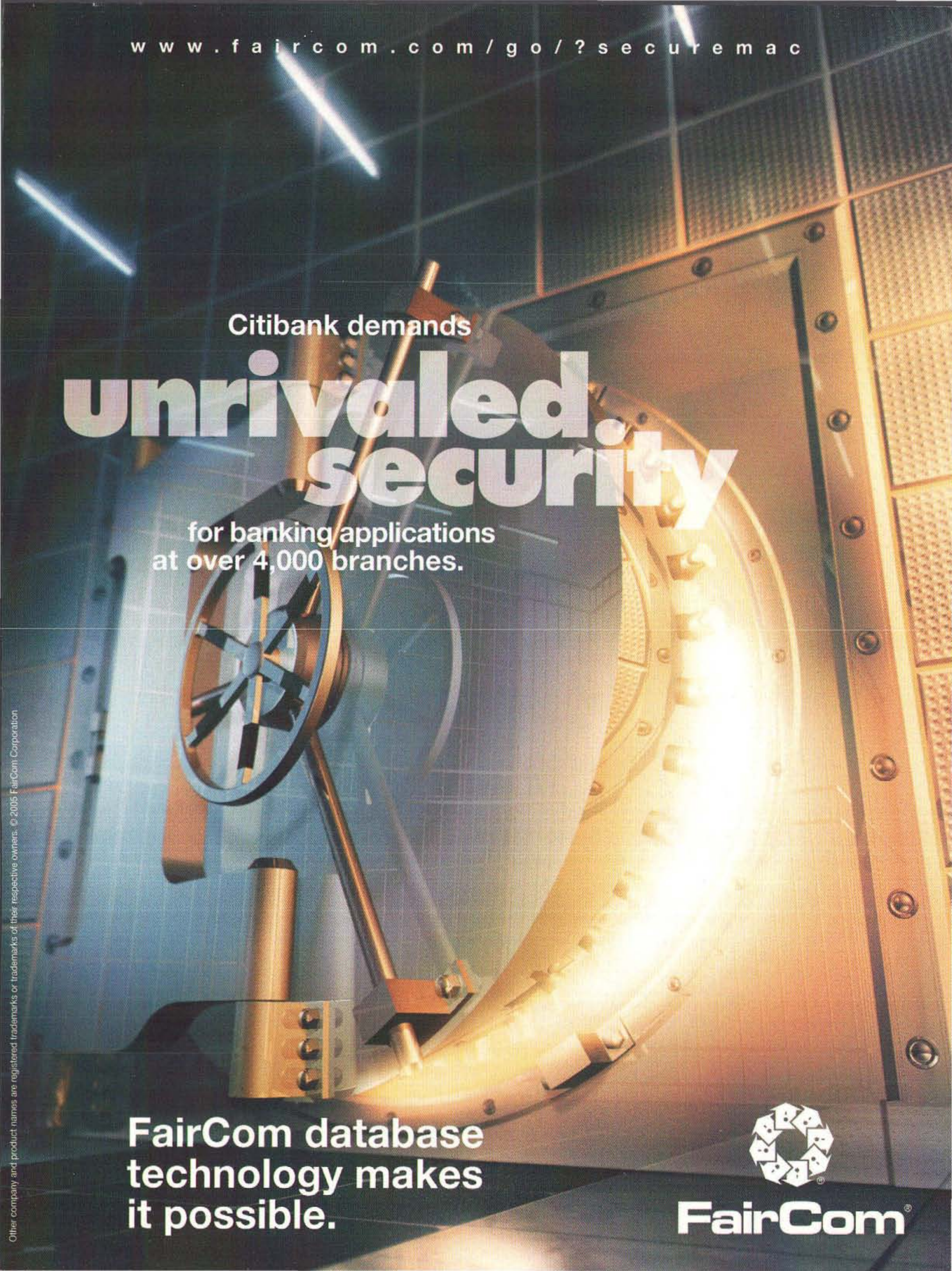
Setting up Nagios 2.0

Writing a Menulet Extension

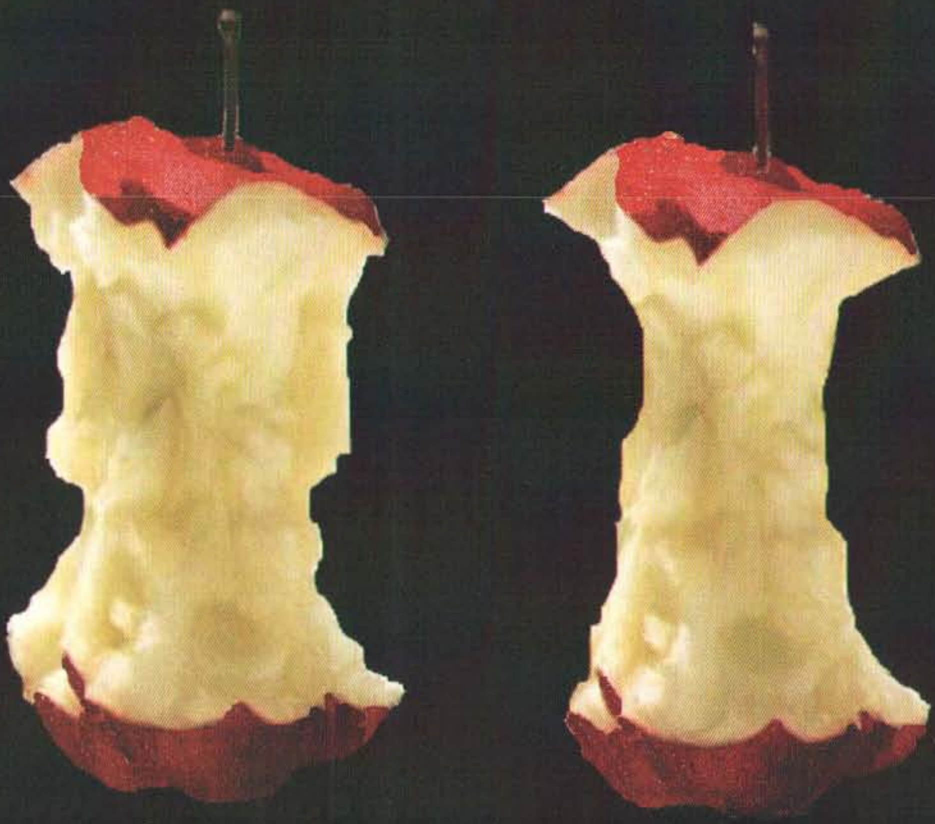
A Look At Virginia Tech's Supercomputer

by Emmanuel Stein

$8.95 US,
$12.95 Canada
ISSN 1067-8360
Printed in U.S.A.


www.faircom.com/go/?securemac
Citibank demands
unrivaled security
for banking applications at over 4,000 branches.
FairCom database technology makes it possible.
FairCom

Really fast compilers are coming ...



Get the most juice out of your new apple cores

New compilers, debuggers, libraries, and powerful tools are coming soon from Intel® and Absoft® for Apple® computers using Intel® processors.

Apply to participate in beta program now!
http://www.absoft.com/trialbeta.html

absoft®

Core technology for Apple developers since 1984
sales@absoft.com 248-853-0050

TABLE OF CONTENTS

ARTICLES & DEPARTMENTS

Multiple Formats. Multiple Platforms. Complex Installers.

We have the solution:

StuffIt Engine SDK

Solving the compression & multi-platform puzzle!
www.stuffit.com/sdk/

Put the power of StuffIt to work for you.

- Adds value to your applications by integrating compression and encryption tools.
- The only tool that supports the StuffIt file format.
- Make self extracting archives for Macintosh or Windows
- Available for Macintosh, Windows, Linux or Solaris

Licenses start as low as $99/Yr.

StuffIt InstallerMaker

An OS X native version ready for developers!
www.stuffit.com/installermaker/

Give your software a solid beginning

- Create Macintosh OS X and Macintosh Classic compatible installers
- Get all the tools you need to install, uninstall or update your software in one complete package
- Add muscle to your installers by customizing your electric registration form to include surveys and special offers.

Prices start at $250

Allume Systems
www.allume.com
email: dev.sales@allume.com

Communicate With Us

Department E-Mails

Orders, Circulation, & Customer Service
cust_service@mactech.com

Press Releases
press_releases@mactech.com

Ad Sales
adsales@mactech.com

Editorial
editorial@mactech.com
(Authors only, no pr)

Accounting
accounting@mactech.com

Marketing
marketing@mactech.com

General
info@mactech.com

Web Site
http://www.mactech.com

In this electronic age, the art of communication has become both easier and more complicated. Is it any surprise that we prefer **e-mail?**

If you have any questions, feel free to call us at 805/494-9797 or fax us at 805/494-9798.

If you would like a subscription or need customer service, feel free to contact MacTech Magazine Customer Service at 877-MACTECH

We love to hear from you! Please feel free to contact us with any suggestions or questions at any time.

Write to letters@mactech.com or editorial@mactech.com as appropriate.

MACTECH®
The Journal of Macintosh Technology

A publication of **XPLAIN** CORPORATION

The Editorial Staff

Publisher & Editor-in-Chief: Neil Ticktin
Associate Publisher & Editor: David Sobsey
Editor-at-Large: Dave Mark
Business Editor: Andrea Sniderman
Production: Dennis Bower

Xplain Corporation Senior Staff

Chief Executive Officer: Neil Ticktin
President: Andrea J. Sniderman
Accounting: Marcie Moriarty
Customer Relations: Susan Pomrantz
Board of Advisors: Steven Geller, Alan Carsrud

Regular Columnists

QuickTime ToolKit: by Tim Monroe
Patch Panel: by John C. Welch
The Source Hound: by Dean Shavit
Reviews/KoolTools: by Michael R. Harvey
AppleScript Essentials: by Ben Waldie
Mac In The Shell: by Ed Marczak

Board of Advisors

Chairman: Dave Mark
Jordan Dea-Mattson, Steven Geller,
Bruce Friedman, and Richard Kimes

Contributing Editors

Michael Brian Bentley, Gordon Garb, Vicki Brown, Chris Kilbourn
Marshall Clow, Rich Morin, John C. Daub, Will Porter, Tom Djajadiningrat
Avi Rappoport, Andrew S. Downs, Cal Simone, Steve Sisak

MacTech Magazine (ISSN: 1067-8360 / USPS: 010-227) is published monthly by Xplain Corporation, 850-P Hampshire Road, Westlake Village, CA 91361-2800. Voice: 805/494-9797, FAX: 805/494-9798. Domestic subscription rates are $47.00 per year. Canadian subscriptions are $59.00 per year. All other international subscriptions are $97.00 per year. Domestic source code disk subscriptions are $77 per year. All international disk subscriptions are $97.00 a year. Please remit in U.S. funds only. Periodical postage is paid at Thousand Oaks, CA and at additional mailing office.

POSTMASTER: Send address changes to **MacTech Magazine**, P.O. Box 5200, Westlake Village, CA 91359-5200.

MacEnterprise, and Behind the Macs at Yale

Interviewing Philip Rinehart: I spoke with Philip over the phone somewhere in the swamps of Jersey (with apologies to Bruce Springsteen). Philip Rinehart is co-chair of the steering committee leading the Mac OS X Enterprise Project (macenterprise.org) and manages Macs as a support specialist at Yale University.

Schoun Regan: Tell me about your job.

Philip Rinehart: I support the Macintosh computers that students use and whatever software and is on them. I'm responsible for the care and feeding of them if you will so that anyone can sit down at the box with a relative degree of confidence that one, it will work, and two, that it will be secure.

SR: What about MacEnterprise?

PR: The goal of the project is for Mac IT professionals to communicate and share information relating to the deployment of Macs in the enterprise and really talk about what it means to drop it into an Active Directory environment or a Netware environment. It's a pool of knowledge...that fosters collaboration between MacIT professionals.

SR: Is it primarily education based?

PR: It started out as...primarily education based...but we're trying to get more enterprise involved. Part of it is that enterprise does not have large Mac deployments relative to education, and typically people who are doing these sorts of things keep their heads down and stay off the radar so they don't get hammered for running Macs. People are posting from all sorts of businesses and we are trying to allow a way for them to communicate.

SR: This gives your more clout as a group with Apple when an issue arises.

PR: That's precisely it. One of the things we want to launch is a bug tracking database which allows people to do the "me too" factor. For example if I post a bug about ***launchd***...then others can chime in. So then a problem that affects me with only $x million in Macs now affects more than $25 million. It means a lot more to Apple Marketing.

SR: The collaboration is really the key

PR: Exactly. This then becomes a priority one instead of a priority five.

SR: How would one join MacEnterprise?

PR: Anybody is welcome. You can have a deployment as small as 10 or as large as 10,000, we pretty much don't care. To a certain extent, if you think about it, the difference between 10 and 10,000 isn't as large as one might think. The issues are essentially the same. We like to get the smaller folks in because it gives them a voice, which they may not otherwise have.

SR: Let's talk tech for a moment. What's the easiest way to shoot an image out back to other Macs?

PR: Use...either command line asr...or Mike Bombich's NetRestore. They use a disk image and they bump it down to the drive.

SR: So I know on a PC using Ghost from Symantec, the way the license fees are structured, it's horrible in my opinion. And Apple's?

PR: As long as you own the operating system and have a site license, you can do what you want. Now that asr is part of the OS, it does not cost you a penny.

SR: What about Radmind?

PR: I'm a user of it myself. Let's say you have an asr image from six months ago, you can take that image and make some changes, test it, and roll back if necessary to that perfect machine state.

SR: Bingo. The word for today is Rollback.

PR: And that to me is the biggest power of Radmind. If you screw something up, you can roll back to the perfect image.

Text Recognition Pen Scanner

IRISPen express

Extremely fast and incredibly accurate, the IRISPen Express is the most effective solution for your retyping needs!

It instantly encodes text and numbers into ANY application - such as word processors, e-mail software, on-line databases, spreadsheets, accounting software, etc. What you get is real editable text, right at the cursor position!

Save time and even enjoy your usual tedious data-entry!

Fun and fast, the IRISPen Express works just like a highlighter. Slide the pen over printed text and numbers, they are automatically retyped at up to 1.000 characters per second!

Smart, Portable, Versatile

The IRISPen Express is light, compact and portable. It installs in 3 min. Use it with any laptop and desktop, wherever you go, wherever you are!

MacTech Special

$99 instead of $129, order today at www.irisusa.com/mactech or call 1 800 447 4744

www.irisusa.com
info@irisusa.com

I.R.I.S.
Document to Knowledge™

SR: C'mon, your users never screw something up.
PR: (Laughs out loud for a long time) Riiiiiiiiiight.
SR: What scripting language is used by most of the group?
PR: Shell
SR: AppleScript?
PR: That too. The combination of the two and Automater is exciting. I think Automater has the potential to do some very cool things.
SR: And scripting for you personally?
PR: I'm a huge Python fan. I used to use PERL but now I love Python.
SR: So what do you say to the high school teacher who wants to lock down the computers for their students? Do they need to know the command line?
PR: There are two ways off the top of my head. If you are using some sort of non-directory setup, you can use the Parental Controls. The better way to do this would be using a directory structure like Open Directory, which allows much more flexability on controls.
SR: If Apple were to call you up and say, "Philip, what one thing can we change on Mac OS X Server to make it better?"
PR: Take Workgroup Manager and give it a complete overhaul. Since half my life is spent in Windows, directory management is a lot easier to use than Workgroup Manager. There are some paradigms that just don't work. Delegated OUs. I've seen articles on AFP548 and Michael (Bartosh) has sent me information on this. I think they (Apple) have made it better but...want more flexibility.
SR: Same question on Client.
PR: Hmmmmmmmm. I would like to see an integrated groupware solution. I think the Outlook interface went out about 10 years ago. I think Apple could take their designers and come up with an excellent interface for it. Mail, iCal, Address Book are nice, but it needs to be better integrated.
SR: What about Windows Server?
PR: That's a tough one. (Pauses) If Apple had better policy based management. Leverage institutional policies and group policies.
SR: How would you walk into a Windows based company and sell them on Macs for their desktop machines. Convince them to swap their Windows desktops with Mac desktops.
PR: To me, it's TCO. It's much lower on the Mac side.
SR: Why?
PR: Initial cost might be more but continued support will be lower. I know that licensing comes into play too. One of the other things I think about is the virus/malware/adware problem. The cost of buying and supporting the software to rid yourself of these problems, costs money. Those costs are not looked at is direct costs, and they should be. You will likely buy Windows Server and now you have more licenses.
SR: I think what happens sometimes is that software purchasing after the fact is not rolled into the initial cost. If IT managers can swoon over low computers prices, then once they arrive, the "I need this and this and this" is now too expensive NOT to implement, because the computers are already there.
SR: Last question. What are your top five favorite movies? Movies that you could watch over and over and over again. No order is necessary, just the top five.
PR: This is tough (laughs): The Third Man, Citizen Cane, Rushmore, Midnight Cowboy, Apocalypse Now.
SR: Philip, thanks very much for speaking with us.
PR: No problem Schoun.

MT

Vital Stats

Years in IT industry : 9
Spoke at Macworld 2004
Information: Philip Rinehart is co-chair of the steering committee leading the Mac OS X Enterprise Project (macenterprise.org) and manages Macs as a support specialist at Yale University. He has been using Macintosh Computers since the days of the Macintosh SE, and Mac OS X since its Developer Preview Release. He presented on OS X security at MacWorld 2004, and is an active member of the Mac OS X Lab project. He has contributed to many areas of the project. Before coming to Yale, he worked as a Unix system administrator for a dot-com company.
Computers: All flavors
Programming Languages: Shell Scripting, AppleScript

About The Author

Schoun P. Regan is CEO of ITInstruction.com, which specializes in Mac OS X training and consulting. He speaks regularly to CEOs and CFOs on how to control IT department spending, the myths surrounding cross-platform integration, and the lunacy of expected lost revenue stemming from a culture bred to tolerate IT staff and operating system inadequacies as "normal". He seeks to change self-fulfilling IT departments that breed complacency for their jobs and contempt for the end user, neither of which are conducive to business.

To: IT Department
From: Graphics Department
Date: Today 9:30 AM
Subject: HELP!

We need you to find a solution that is going to track and archive our production jobs and files from start to finish. And make them easy to find after we archive them. Even if our files are offline. And it needs to integrate with our **FileMaker Pro database**...

... by tomorrow.

Relax, We've Got You Covered...

Introducing

Digital*StorageManager*™ 1.0

Intranet Search and Archive Management for Creative Workgroups

Digital Storage Manager is a high-capacity, high-performance file search and archive management system for creative, prepress and publishing workgroups. Deployable as either a stand-alone product or an integrated solution for FileMaker Pro, Digital Storage Manager provides real-time indexing of OS X and Windows file servers, automated metadata tagging, and powerful archiving capabilities.

Meta COMMUNICATIONS

Download a 21 Day Trial or Call to Schedule a Demo

Web: www.meta-comm.com Tel: 1-800-771-6382 Email: sales@meta-comm.com

TIMELINE

MAPPING THE EVOLUTION OF QUICKTIME PROGRAMMING

The QuickTime programming landscape looks pretty good nowadays. In terms of what you can do with QuickTime, the story has never been better. In the nearly 15 years since its introduction in December 1991, QuickTime has gained a truly impressive set of multimedia capabilities. It now provides services for displaying and creating movies, capturing audio and video data, compressing and transcoding media data, broadcasting saved movies and live captured data across a network, displaying and modifying still images, and other tasks too numerous to list here exhaustively. With the recent appearance of QuickTime 7, the QuickTime programming APIs now comprise more than 2500 actively-supported functions.

Introduction

But that of course is only half of the story, and in many ways it is the less interesting half. We would expect this sort of continual feature expansion from any software architecture that has been around for a decade and a half. What is perhaps more interesting is the wealth of tools that we as software developers can use to access those multimedia capabilities. For almost three full years, believe it or not, this ***QuickTime Toolkit*** column has focused more or less directly on the issue of how to use various different programming languages and development environments to construct QuickTime applications. We've built applications using a wide variety of alternate languages and IDEs, including Cocoa, REALbasic, Revolution, Visual Basic, AppleScript Studio, Java, Tcl/Tk, and others. From modern object-oriented application frameworks to old-school scripting languages, we've pretty much run the gamut of possibilities for developing applications and tools to create and modify and display QuickTime content.

It would be nice to pause and reflect on these tools and languages and to see how they compare with one another in terms of ease-of-use and feature completeness and extensibility. It would also be nice to run some benchmarks to see if some of these development environments produce particularly more efficient and resource-friendly applications than others. (Java, for instance, has a reputation for being slow; it would be nice to actually test our sample Java-based player application against our other sample applications.) But those reflections will have to wait for some other opportunity, since in this article I want to discuss a somewhat different issue. In particular, I want to look at how QuickTime programming itself has evolved in the years since its introduction. What did it look like in the beginning, and what is its general character now? What sorts of forces have prompted changes in the QuickTime programming model?

I think that this is an interesting set of questions because not every QuickTime developer — and in fact probably a minority of current QuickTime developers — has been using QuickTime for a significant portion of those 15 years. In addition, most developers are probably using one or more of the QuickTime-savvy RAD tools or application frameworks.

Since none of these tools or frameworks provides access to all the existing QuickTime capabilities, it's likely that some QuickTime developers will need to venture outside the limits of their chosen tools to develop plug-ins or libraries for those tools. And then they land squarely in the realm of those 2500 functions.

MacOS

So let's begin at the beginning. QuickTime was originally released on the Macintosh Operating System (specifically, on MacOS version 6.0.7). Quite sensibly, the original QuickTime APIs were heavily dependent on the data types and structures used by the Macintosh Operating System and the Macintosh User Interface Toolbox. A chunk of memory was typically specified using a `Handle` data type, and files were typically specified using `FSSpec` records. Data to be drawn on the screen was accessed using bitmaps drawn into graphics ports and graphics worlds (specified using `GrafPtr` and `GWorldPtr` data types). The intention was very clearly that the QuickTime APIs should fit into the existing programming model on Macintosh computers.

At the same time, the QuickTime architects did not hesitate to drive that programming model forward in certain important ways. One of the big departures from existing practices was to make C the language of choice for developing QuickTime applications, in spite of the fact that Pascal still dominated MacOS software development during the time QuickTime was being developed. The original developer CD for QuickTime 1.0 provided 18 sample projects using C but only half that many using Pascal. More importantly, the technical documentation for QuickTime provided all sample code and reference material in C, not Pascal. Indeed, the books ***Inside Macintosh: QuickTime*** and ***Inside Macintosh: QuickTime Components*** were the very first books in that series to relegate Pascal to the programming summaries at the end of the chapters.

Listing 1 shows what a typical routine to open a movie file might have looked like. It uses the Standard File Package to display the file-opening dialog box to the user, and then it calls `OpenMovieFile` and `NewMovieFromFile` to create a `Movie` identifier for the data in the movie file.

Listing 1: Loading a movie from a file

```
Movie GetAMovie (void)
{
  OSErr                  myErr;
  SFTypeList             myTypes = {MovieFileType, 0, 0, 0};
  StandardFileReply    myReply;
  Movie                  myMovie = NULL;
  short                  myRefNum;
  short                  myResID = 0;

  StandardGetFilePreview(NIL, 1, myTypes, &myReply);

  if (myReply.sfGood) {
    myErr = OpenMovieFile(&myReply.sfFile, &myRefNum,
                              fsRdPerm);
    if (myErr == noErr) {
      NewMovieFromFile(&myMovie, myRefNum, &myResID, NULL,
                                   newMovieActive, NULL);

      CloseMovieFile(myRefNum);
    }
  }

  return myMovie;
}
```

One interesting thing about this code is that it is almost completely deprecated on current Macintosh computers. The Standard File Package never made the jump from the "classic" MacOS to Mac OS X, and (as we saw in the previous article, "State Property 2" in ***MacTech***, December 2005) the `NewMovieFromProperties` function is now recommended in place of `NewMovieFromFile`. After all, the parameters to `NewMovieFromFile` include oddities like a file reference number and a pointer to a resource ID, which are not standard ways of accessing files or file data on Mac OS X.

It's worth remarking that first QuickTime developer CD also included a small set of HyperCard add-ons, called ***external commands*** or ***XCMDs***, that allowed HyperCard developers to access QuickTime functionality in their stacks. This then marks the first integration of QuickTime into what might be called a rapid application development (RAD) tool. There were four XCMD modules:

(1) The `QTMovie` XCMD, which could be used to play QuickTime movies in a window or directly onto the screen;
(2) The `QTRecordMovie` XCMD, which displayed data from a video digitizer;
(3) The `QTEditMovie` XCMD, which supported editing operations on a QuickTime movie;
(4) The `QTPict` XCMD, which performed a variety of still image operations, including displaying a picture on a card, compressing pictures, and allowing control over the clipping region of the card window.

(The perceptive reader will notice that, by pure historical accident, one of these XCMDs shares its name with the principal class in the new Cocoa QTKit framework, `QTMovie`.)

Windows

In the early 1990's, Apple released a version of QuickTime (called "QuickTime for Windows") that provided support for playing QuickTime movies on Windows computers. While it was a significant step forward, this version had some severe limitations. Most importantly, it provided a playback engine only; there was no way to create QuickTime movies on the Windows platform. Also, many of the APIs for playing movies back differed from their Macintosh counterparts. For instance, on the Mac, `NewMovieController` is declared essentially like this:

Long Distance

2.9¢ Per Minute!

Straight 6 second billing increments

Excellent rates on intrastate, intralata/toll calls and international calling with no term contract.

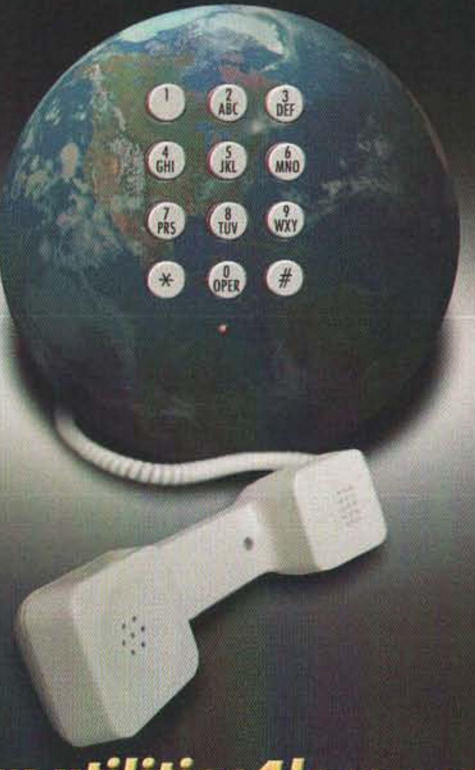
Toll Free (800/888/877/866) service, same low per minute rate for incoming calls.

10 cents per minute calling card.

Detailed billing directly from OPEX.

Quality electronic and telephone customer support. No monthly billing fee if your bill is over $20.00 each month.

(NOTE: $2.00 billing fee is charged when your bill is under $20.00.)



www.utilities4less.com

```
MovieController NewMovieController (Movie theMovie,
                    const Rect *movieRect, long someFlags);
```

But under QuickTime for Windows, it had this declaration:

```
MovieController NewMovieController (Movie theMovie,
                    const LPRECT lprcMovieRect, long someFlags,
                    HWND hWndParent);
```

You'll notice that the Windows version took an additional parameter (`hWndParent`) and that the type of the second parameter was a pointer to the standard Windows rectangle type (`RECT`), not the Macintosh rectangle type (`Rect`).

QuickTime 3.0, released in 1998, changed all that. It provided a set of APIs that were virtually identical — in both parameter lists and feature completeness — on Macintosh and Windows platforms. It was then possible to write Mac and Windows applications that used the same source code, at least for the QuickTime-specific portions of the application.

The magic provided by the Windows version of QuickTime 3.0 was accomplished principally by a library called the ***QuickTime Media Layer*** (or, more briefly, ***QTML***). The QuickTime Media Layer provides an implementation of a number of the parts of the Macintosh Operating System (including the Memory Manager and the File Manager) and the Macintosh User Interface Toolbox (including the Dialog Manager, the Control Manager, the Resource Manager, and the Menu Manager). In other words, QuickTime was ported to Windows mainly by way of transplanting large portions of system software from the MacOS to Windows.

For existing Macintosh developers, this scheme had some profound benefits. First and foremost, this greatly reduced the need to learn the intricacies of a new operating system. To display the standard Windows file-selection dialog box to elicit a movie file from the user, a developer could just use the familiar `StandardGetFile` function that he or she had been using all along on MacOS. And custom application icons, sounds, and fonts could be stored in resources, just as they are with MacOS applications. And existing QuickTime code could, as noted above, simply be recompiled for Windows applications. (Indeed, the code in Listing 1 would ***still*** compile and link just fine on Windows computers.)

But for Windows developers, this scheme was less than optimal. It required working with unfamiliar data types, like `Handle` and `FSSpec` and `GrafPtr`, and also working with command-line tools to create resources or add them to application files. A better solution, which Apple and several third-party developers pursued, was to develop Component Object Model (COM) plug-ins that support QuickTime APIs in Windows applications. One type of COM object is an ***ActiveX control***, which can display a user interface and process events directed at that interface. The developer can then support QuickTime in a COM-aware application (for instance, one developed using Visual Basic) by using an appropriate ActiveX control. For instance, Listing 2 shows some Visual Basic code to handle the Open menu item in the File menu.

no more auction management headaches.

sellitol bestrate 500mg

INDICATIONS: For rapid relief from monthly fees, disorganization, and lost sales. Aids in listing, inventory management, email marketing, trends analysis, customer retention management, and other activities associated with selling profitably on eBay.

DOSAGE: One installation. Subsequent updates are automatic.

DIRECTIONS: List. Sell. Repeat.

ACTIVE INGREDIENTS: automation. scheduled events. custom actions. offline capability. import from turbolister, excel, and databases. ledger. groups. cash flow analysis. support for eBay stores and eBay motors. custom reporting. html editor. bulk listing, ending, revision. relisting. free scheduled listing. offline listing preview. ad templates. image management. watermarks. spell check. consignment management. profiles. vendor management. auto reorder. product sales analysis. questions and feedback management. spam control. fraud management. email templates. invoicing. dispute management. email marketing campaigns.

WARNING: Side effects may include increased sales, higher efficiency, and euphoria.

marketblast

please visit www.marketblast.com for a free sample.

Listing 2: Handling the Open menu item

```
Private Sub FileOpen_Click()
  Dim openDial As New DialogWindow
  On Error GoTo bail

  openDial.CommonDialog1.Filter = "All Files (*.*)|*.*|Movie
Files (*.mov)|*.mov|Flash Files (*.swf)|*.swf"
  openDial.CommonDialog1.FilterIndex = 2
  openDial.CommonDialog1.Flags = 4

  'hide the "Read Only" check box
  openDial.CommonDialog1.CancelError = True
  openDial.CommonDialog1.ShowOpen

  OpenFile (openDial.CommonDialog1.FileName)
  Unload openDial
  Exit Sub

bail:

  'the user pressed the Cancel button

  Unload openDial
  Exit Sub
End Sub
```

The `FileOpen_Click` handler uses standard Visual Basic methods, except for the application defined `OpenFile` method, shown in Listing 3.

Listing 3: Opening a movie file

```
Sub OpenFile(fileNm As String)
  Dim movieWind As New MovieWindow

  If Len(fileNm) = 0 Then
    movieWind.Caption = "Untitled"
  Else
    movieWind.Caption = BaseName(fileNm)
    movieWind.QTActiveXPlugin1.SetURL (fileNm)
  End If

  movieWind.Show
End Sub
```

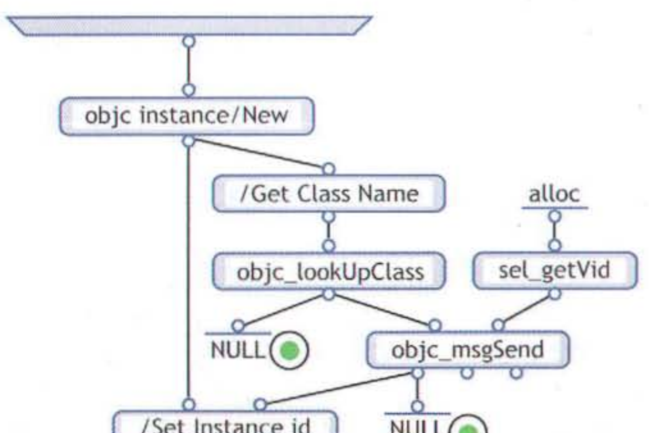
It's important to note that a QuickTime-savvy ActiveX control does not so much remove the dependence on QTML as hide it. That is to say, although the Visual Basic developer doesn't need to know about MacOS data types, the person who wrote the ActiveX control does. And even the VB developer ***might*** need to know about MacOS data types when using the `declare` statement to reference external procedures in the QTML library. This would happen if the developer needs to access QuickTime functionality that was not implemented in whichever ActiveX control he or she is using.

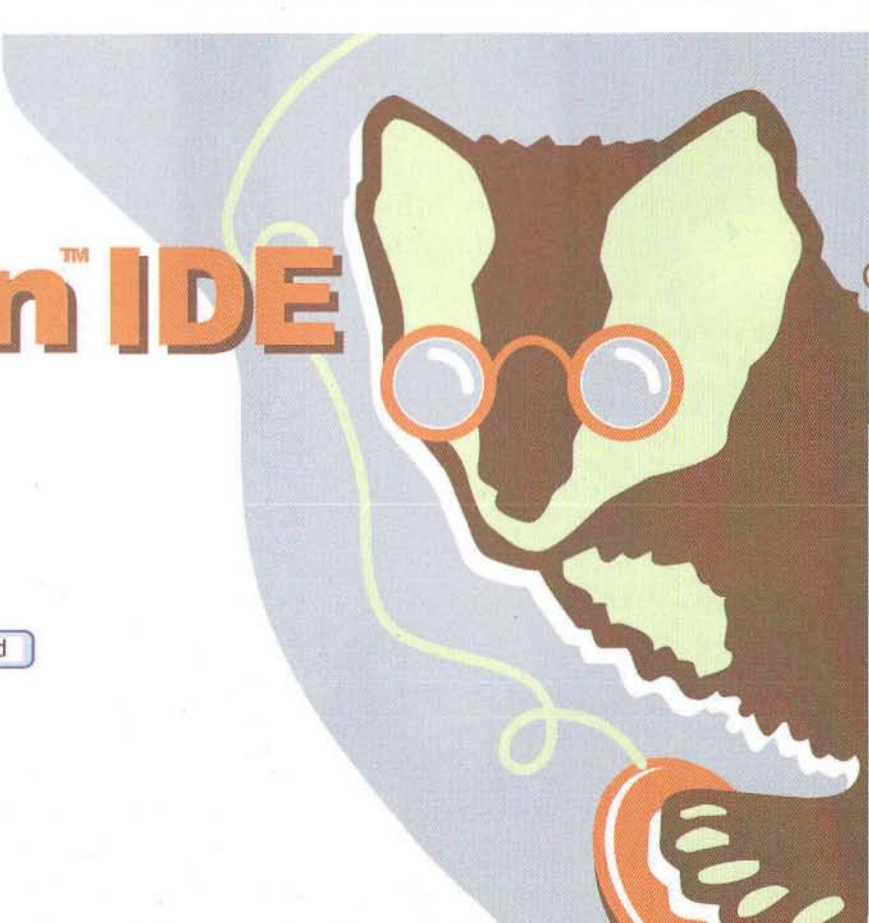
Mac OS X

QuickTime's migration from MacOS to Mac OS X is remarkably similar in spirit to its migration from MacOS to Windows. Once again, a software layer was added to support the Macintosh Operating System and User Interface Toolbox managers that originated on MacOS and which are used extensively throughout the QuickTime source code. Mac OS X is a UNIX-based operating system and provides no more native support for QuickTime than does Windows. In this case, the implementation of the Macintosh Operating System and Toolbox managers is provided by a library called ***Carbon***. The only real difference between QTML and Carbon is that Carbon has

Draw code
You know you want to

Marten™ IDE





• Edit while you run • One click Cocoa and Carbon application creation • Powerful debugger

w w w . a n d e s c o t i a . c o m

evolved more swiftly than QTML. For instance, as mentioned earlier, the Standard File Package has long since been deprecated on Macintosh computers, having been replaced by the Navigation Services (which supports longer filenames and alternate text encoding schemes such as Unicode).

The move to Mac OS X has prompted two additional sorts of changes to QuickTime APIs, above and beyond the changes required for it to keep pace with enhancements in the Carbon library. First, a reasonably extensive Cocoa framework, QTKit, was developed to replace the existing QuickTime-related classes, `NSMovie` and `NSMovieView`. We investigated QTKit in several recent articles (***MacTech***, May, June, and July 2005) and saw that we can develop full-featured Cocoa applications with only minimal need to venture outside of the methods it provides. And venturing outside of Cocoa is easy, because Objective-C is a superset of ANSI C. This means that we can easily call Carbon APIs within our Cocoa code. For example, Listing 4 shows a method for setting the magnification level of a Flash movie opened using QTKit. Notice that we call `GetMovieIndTrackType` to find the first Flash track in the movie, and then we call `GetTrackMedia`, `GetMediaHandler`, and `FlashMediaSetZoom` to set the zoom level of that track.

Listing 4: Setting the zoom level of a Flash movie

```
- (void)setZoom:(float)zoomPct
{
  Track flashTrack = NULL;
  Media flashMedia = NULL;
  MediaHandler flashHandler = NULL;

  flashTrack = GetMovieIndTrackType([self quickTimeMovie],
                 1, FlashMediaType, movieTrackMediaType |
                 movieTrackEnabledOnly);
  if (flashTrack) {
    flashMedia = GetTrackMedia(flashTrack);
    flashHandler = GetMediaHandler(flashMedia);
    FlashMediaSetZoom(flashHandler, zoomPct);
  }
}
```

The second principal way in which Mac OS X has affected QuickTime, on the API level, is the adoption within QuickTime of Core Foundation data types. ***Core Foundation*** is a procedural C framework that is modeled on the object-oriented Foundation framework in Cocoa. It provides, among other things, some very nice collection classes (such as arrays and dictionaries) and Unicode-compatible strings. QuickTime 6.4 introduced, for instance, several functions for creating data references from Core Foundation data types like `CFString` and `CFURL`, including these:

```
QTNewDataReferenceFromFullPathCFString
QTNewDataReferenceFromURLCFString
QTNewDataReferenceWithDirectoryCFString
```

And we saw in an earlier article ("State Property" in ***MacTech***, November 2005) that the QuickTime property function `QTGetMoviePropertyInfo` can return reference-counted Core Foundation objects that need to be released (by calling `CFRelease`). Interestingly enough, these Core Foundation data types and functions are supported now on Windows as well as on Mac OS X.

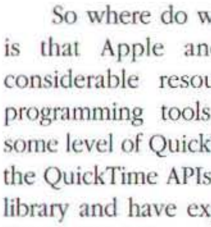
Conclusion

So where do we stand here in early 2006? The good news is that Apple and third-party developers have invested considerable resources into making sure that the major programming tools and development environments support some level of QuickTime movie playback and editing. Moreover, the QuickTime APIs have kept pace with changes in the Carbon library and have expanded to provided support for Cocoa and Core Foundation programming paradigms.

The bad news, if there is any, is that some important parts of QuickTime are still accessible only using functions and data types are arose on MacOS, a now-deprecated operating system. It would be nice to never have to allocate another `Handle` object. We aren't quite there yet. But surely some day we will be.

MT

About The Author



Tim Monroe is a member of the QuickTime engineering team at Apple. You can contact him at monroe@mactech.com. The views expressed here are not necessarily shared by his employer.

professionalsupportrobus
bustnesseaseofuseperform
easeofuseWHYperforman
ormanceMPI/PROscalabili
sscalabilityFORprofession
robustnessMACrobustness
tyeaseofuseOSscalabilityp
performanceX?professiona
alabilityprofessionalsuppo
tnessperformancescalabili

For more information on MPI/PRO please visit www.verarisoft.com

If you would like to see the performance for yourself e-mail us at sales@verarisoft.com for a trial evaluation

Verari Systems Software, Inc., 110 12th Street, Suite D103, Birmingham, AL 35203, USA
PHONE: 1-205-397-3141 • FAX: 1-205-397-3142
http://www.verarisoft.com/
The Verari Systems Software Distinctive Logo is a trademark of Verari Systems Software, Inc. All images, text, and content © 1999-2006, Verari Systems Software, Inc.

Verari systems software™

Introduction to Core Data, Part IV

Storing Fetch Requests in your Data Model

By Jeff LaMarche

In the last Core Data article we talked about creating Fetch Requests programmatically using format strings and predicates. Using format strings to create Fetch Requests is a little bit inelegant, although it does give you a tremendous amount of flexibility. For your day-to-day, run-of-the-mill data fetching needs there is an easier way that gets those format strings out of your compiled code. You can create Fetch Requests right in your data model. This month, we're going to take a quick look at doing just that: We're going to build Fetch Requests and predicates right in Xcode and then look at how we get those Fetch Requests from our code.

Introduction

In Xcode, I went ahead and built a simple data model with a single entity called ***Person*** that contains two string attributes called ***firstName*** and ***lastName***, and a date attribute called ***birthDate***. I also gave the entity a to-one relationship called ***spouse*** that can connect this entity to one other ***Person*** entity. I'll be using this simple data model to illustrate my points in the rest of this article. We've already discussed creating data models back in the first part of this series (***Intro to Core Data,*** July, 2005), so I won't be guiding you through creating the data model this month. If you create a new project in Xcode using the ***Core Data Application*** project template, you should be able to quickly create the same data model yourself, or you can simply download my project from the MacTech FTP site. You might want to do that before you continue with the article, as I'll be making frequent references to this data model, however it is a simple enough data model that you can probably follow along without it.

Stored Fetch Requests

If you search Apple's documentation for information on how to store Fetch Requests in your data model, you might come to the conclusion that it simply can't be done. The ability to store Fetch Requests as part of the data model, although present in Core Data's predecessor Enterprise Objects Framework, seemed to be absent from the data model editor in Xcode when Core Data was first released. It was actually there, but it just wasn't very obvious; you had to change the view settings for the properties pane in order to see or work with Fetch Requests (Figure 1). The official documentation spends so little time on the topic that it's very easy to miss it.

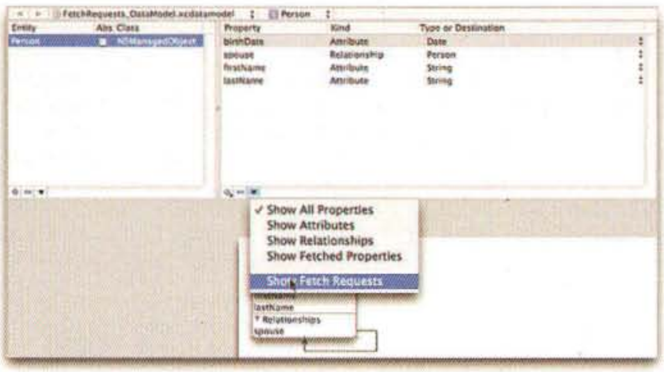


Figure 1. Getting to the fetch requests stored in your data model.

With the recent release of Xcode 2.1, however, the eagle-eyed among you have probably noticed that a fourth option has been added to the property pane's ***add*** (plus sign) pop-up menu in the data model editor (Figure 2). Unlike with earlier releases, this option now shows up regardless of whether you've changed the pane to show Fetch Requests or not. That makes the functionality a little more obvious, and a little easier to access.

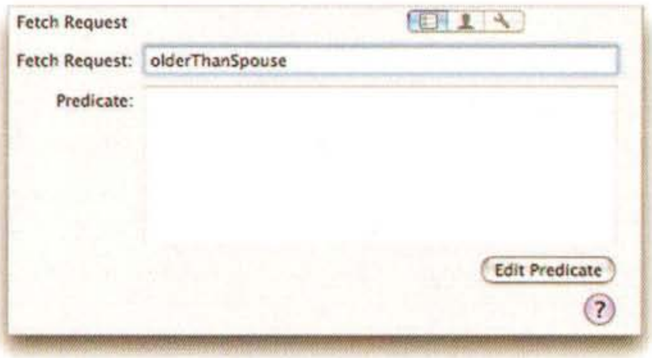
Figure 2. You can now add a fetch request when looking at the properties pane in any view mode.

If you decide to store fetch requests in your data model (and you should), you still have to know about predicates and format strings. All that information I gave you in the Core Data III article back in November was not for naught. Without a solid understanding of format strings and predicates, you wouldn't be able to use the functionality we're going to go over this month. See? There is a method to my madness.

The Predicate Builder

The way that we build Fetch Requests in Xcode is to use the ***Predicate Builder***. This is a tool that is built into Xcode's data model editor, and is used for visually creating predicates (and the format string that they're based on). Why don't you go ahead and select the ***Person*** entity in your data model, then press the little plus sign in the properties pane (the top-middle pane). Select `Add Fetch Request` from the menu that pops up (Figure 2). You'll get a new Fetch Request, which you should go ahead and rename to ***olderThanSpouse*** in the top right pane. We are going to create a predicate that brings back all people who have a spouse and are older than them.

When you edit the name of this Fetch Request, you should see a button labeled ***Edit Predicate*** down below where you change the name. This button will call up the Predicate Builder. The large text field above the ***Edit Predicate*** button will show you the format-string associated with any predicate you build with the Predicate Builder. Although it looks like you should be able to edit the format string there, you actually can't, though you can select it and copy to the clipboard.


Fetch Request

Fetch Request: olderThanSpouse

Predicate:

Figure 3. You can get to the predicate builder by pressing the Edit Predicate button when the fetch request is selected in the properties pane.

Go ahead and click that button to bring up the Predicate Builder. This interface probably looks familiar to you, at least if you've ever set up a rule in Mail, or defined a smart playlist in iTunes or a smart album in iPhoto. Predicates are built using a fairly standard-issue Apple rules interface (Figure 4). Almost. It's a very functional interface, but it's not as intuitive as the ones you've probably used. There's actually a little bit of hidden functionality in there, so let's take a closer look at it.

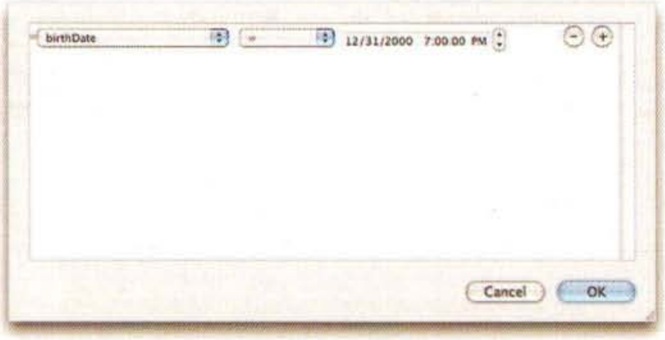

birthDate = 12/31/2000 7:00:00 PM

Figure 4. The Predicate Builder

Starting from the left side, the first thing you'll see is a pop-up menu. This is the ***key pop-up menu***, sometimes simply referred to as the ***left hand side menu***. This is where you define what you're comparing or, in other words, the left side of the format string. Most often, you're going to select one of the attributes of your entity. If you click the pop-up menu, you'll see that you have more options to select from than just the attributes of the current entity, however (Figure 5).

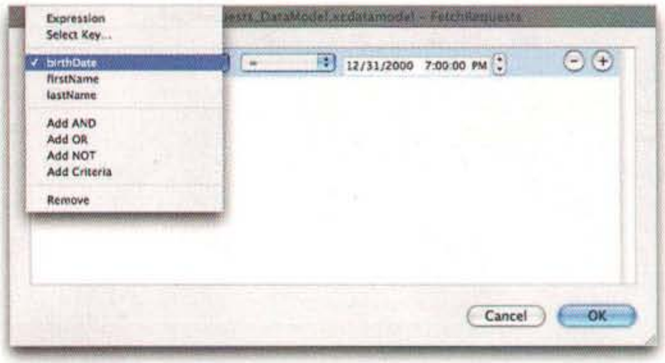


Figure 5. The key pop-up menu.

The top-most item in the menu is labeled `Expression`. You won't find this option well-documented anywhere unfortunately, but it's a very handy one nonetheless. This option changes the whole line to a single editable text field where you can plug in a format string. Why would you want to do that? Well, there might be times when you need to access a property using key-value coding that the data model editor doesn't know about, perhaps because you've created an attribute and left its type as ***undefined***. Basically, if you know about an attribute that Xcode doesn't know about for any reason, you can use this option to manually enter the format string for one part of the predicate. In practice, you probably won't have to use this very often, but it's good to know it's there in case you ever need it.

The next option, labeled `Select Key...` brings up a browser to let you select an attribute. This might seem a little redundant, since the individual attributes of this entity are also available right in the pop-up menu, but you might have noticed that the ***spouse*** relationship was not accessible there, because it's not an attribute. Although this option can be used just to select an attribute of this entity, the primary reason it exists is so that you can drill down into other entities through this entity's relationships. Let's say we wanted to create a fetch request based on the birthday of the person's spouse. In that case, you could use the `Select Key...` option, click on the spouse entry in the left-most column, then click on the ***birthDate*** field in the second column (Figure 6). Don't actually do that, however. Click the ***Cancel*** button; I don't want you to actually select the spouse's ***birthDate*** field here.

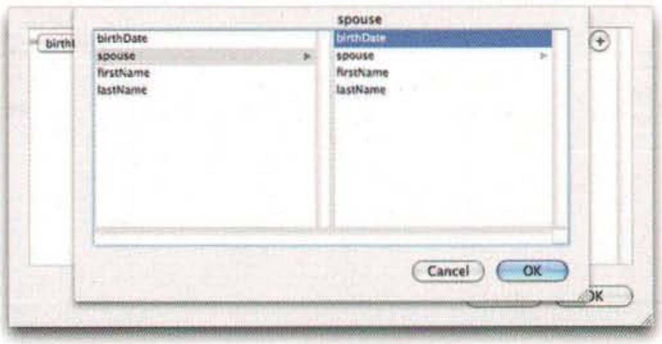
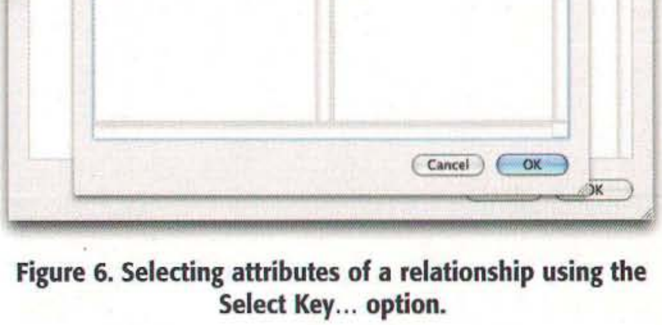


Figure 6. Selecting attributes of a relationship using the Select Key... option.

The next few options are pretty self-explanatory; they are simply the attributes of this entity that you can select (***firstName***, ***lastName***, ***birthDate***) The options in the next section below the attributes allow you to add additional criteria to this rule. Here's where we start to see a deviation from the standard-issue Apple rules dialog. In most rules dialogs, such as iTunes' smart playlist dialog (Figure 7), you define a simple list of criteria and choose to require either all of the criteria, or any single criteria in the list.


Smart Playlist

✓ Match all of the following rules:

Artist contains Gun

Year is less than 1990

☐ Limit to 25 songs selected by random

☐ Match only checked songs

✓ Live updating

Figure 7. The smart playlist rules dialog from iTunes.

The Predicate Builder allows you to create much more complex and robust rules than the rule dialogs in Apple's consumer applications. You can use all three of the standard Boolean operators (and, or, not), and choose a different operator for each pair of operations. You can also set precedence by creating a hierarchy of criteria.

The four `Add` options under the key pop-up menu all insert another criterion into the predicate, and function almost exactly the same. The only difference in the options, is which of the Boolean operators is used, although that can be changed after the fact. The bottom-most `Add` option is the same as pressing the plus button on the right-hand side of the row or the `Add AND` option. You can look at Figure 8 to get an idea of the sort of complex rules you can build using the Predicate Builder. The last option on the left-hand side menu simply removes the row and functions exactly like pressing the minus button on the right-hand side of the row.

Premium

Small Business Management & Accounting

Software

Mind Your Own Business. Smarter.

800.322.MYOB (6962)

www.myob-us.com

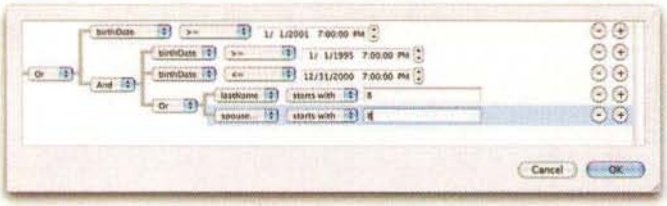


Figure 8. Crafting a complex Fetch Request in the Predicate Builder.

As we move to the right, the next pop-up menu is the operator to be used for this row. The available operators change depending on the type of attribute selected in the key pop-up menu. Different options are presented, for example, when you select a string attribute than when you select a number or date attribute.

Last, but not least, the next field over is simply the value to be compared when evaluating this criterion. The example in Figure 8 shows a number of comparisons being made, but they're all being made to constant values typed right into this field. That's useful, but not always what you want. To be really useful, you'd need to be able to plug in values at runtime or, perhaps, compare one attribute to another. But there doesn't seem to be any way to enter anything other than a constant here, does there?

Well, here's a bit of hidden functionality. Go ahead and either right-click or control-click in the space to the left of the minus button that's used to delete the row. Lo and behold, you'll be presented with a contextual menu with some new options(Figure 9).

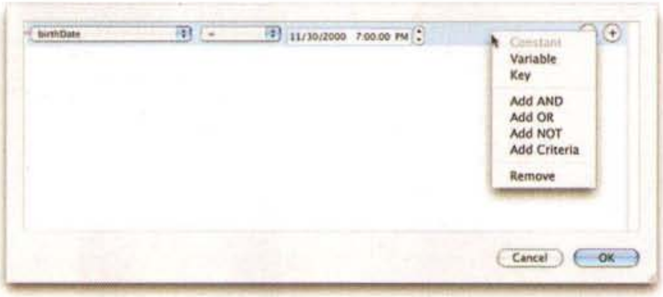


Figure 9. The hidden contextual menu.

The bottom five options are exactly the same as ones you saw in the left-hand key pop-up menu. They add or remove the current row. The top three options, however, are the magic options. They allow you to select the types values for the right-hand side of the equation. The default value, as you've seen, is ***Constant***, which allows you to specify the comparison value right in the Predicate Builder.

Let's skip the second option (***Variable)*** for just a second. We'll come back to it in a moment, but it's more involved than the other two, so let's look at the third option first. Go ahead and select ***Key***. Doing that, will change the place where you used to be able to enter a constant value into another pop-up menu that looks identical to the left-hand side pop-up menu. This is the right-hand side key pop-up menu. This can be useful in many situations. We stated that we were going to create a Fetch Request to retrieve all people who are older than their spouse. Easy enough. Just choose `Select Key…` from the right-hand side pop-up menu and drill down to the spouse's ***birthDate*** as we did a moment ago on the left-hand side. Next, change the operator pop-up menu to greater-than and make sure `birthDate` is selected in the left-hand key pop-up menu. (Figure 10).

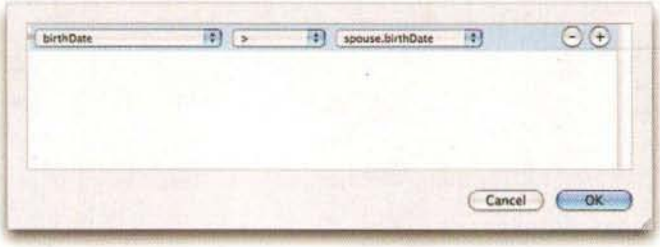
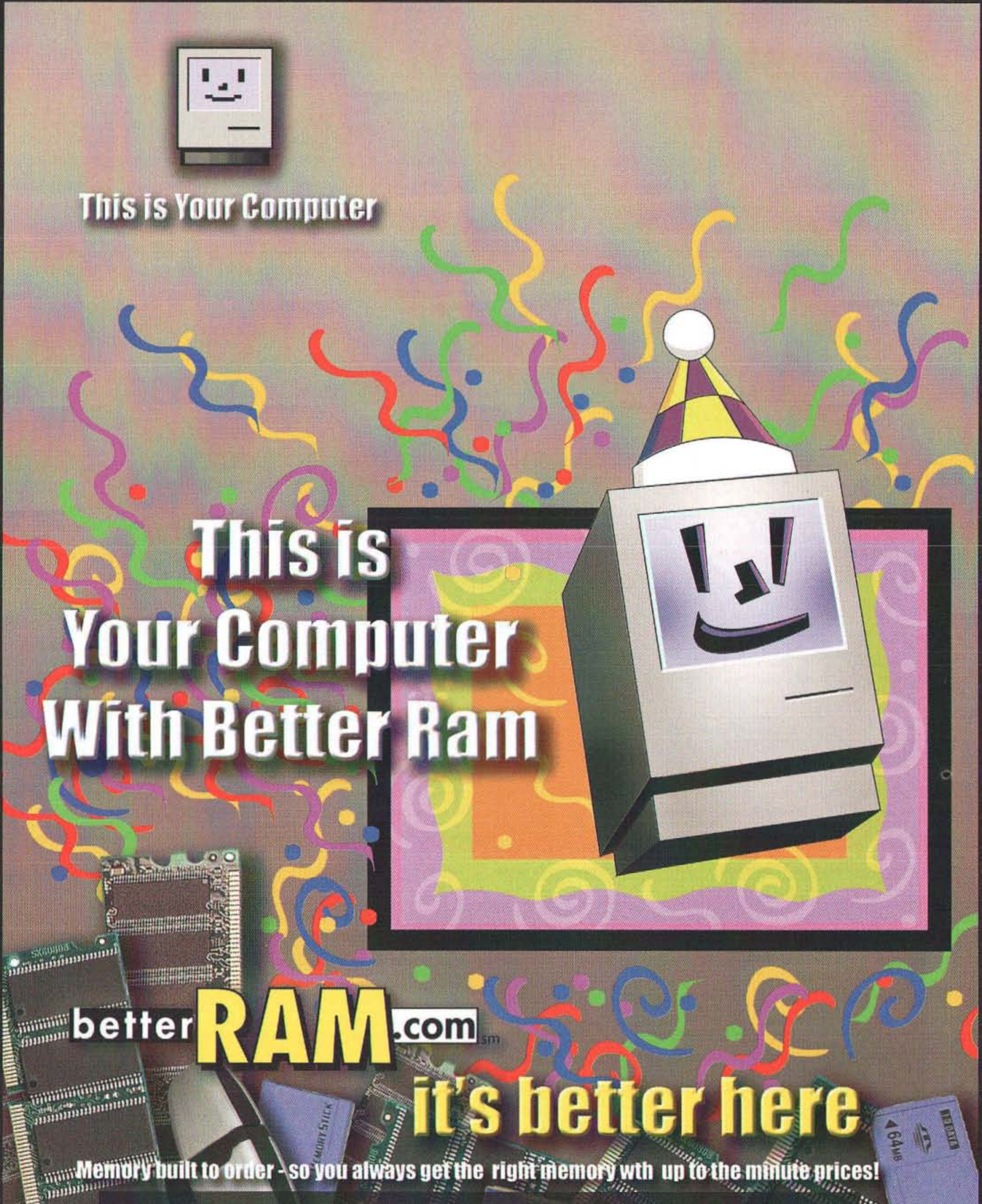


Figure 10. Comparing a key to another key

Go ahead and click the ***OK*** button; this Fetch Request is done. You can take a look in the ***Predicate:*** field to see the format string you just created. Now, let's create a little more complex Fetch Request. Go ahead and click the plus button in the lower left of the Fetch Request pane (the top-middle pane of the data model editor). When you added a Fetch Request before, the Property pane turned into the Fetch request pane, so this time, you can just click the plus button and don't have to select from a pop-up menu. When the new Fetch Request comes up, rename it to ***birthDateBetween***, then click the ***Edit Predicate*** button to get back to the Predicate Builder. We're going to build a Fetch Request to retrieve all people with a birth date between two values, but we don't want to specify those two values until runtime.

Let's now talk about the second option on the contextual menu: `Variable`. If you control-click to the left of the minus button again and select that option, you'll get an editable text field labeled ***Variable*** (Figure 11). This is where the juju happens, baby. In this field, we can specify a variable name and later we'll be able to replace that variable with another value in our code. This is called a ***substitution variable*** or sometimes a ***bind variable***. This is what makes stored Fetch Requests flexible enough to use for just about everything. If you specify a value here – which can be just about any string value you wish to use – you will then be able to substitute any other value of your choosing when you retrieve the Fetch Request from your data model.

If you're playing along at home, go ahead and change the operator pop-up menu from = to `within`, which will add a second row to this criteria. The new row is not a new criteria, but just a second parameter to the existing one. The `within` operator allows you to search based on a date attribute that falls between two other date values. Control-click to the right of the new row that got entered and change it also to `variable`. Now, go ahead and assign each of those variables a name. I chose ***FROMDATE*** and ***TODATE***, as you can see from Figure 11. Once you've got it set up, click the ***OK*** button.


This is Your Computer
This is Your Computer With Better Ram
betterRAM.com sm
it's better here
Memory built to order - so you always get the right memory wth up to the minute prices!
Toll Free: (800) 895-3493 Outside US/Canada: 805-494-9797 Fax: 805-494-9798

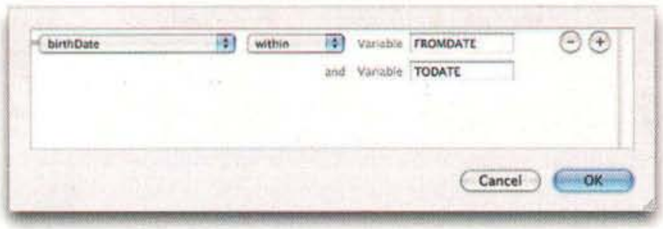


Figure 11. Specifying a substitution variable in Predicate Builder.

After the Predicate Builder is dismissed, go ahead and look at that large text field labeled *Predicate:*. That will show you the format string that will be used for creating this Fetch Request. Notice the dollar sign in front of FROMDATE and THRUDATE? That's how substitution variables are specified in format strings.

Go ahead and save your data model and let's take a look at how we can pull our Fetch Request out of the data model instead of having to create them from scratch, as we did in the previous article.

Getting Fetch Requests Out Of the Data Model

Technically speaking, you aren't storing a Fetch Request in your data model, even though it's usually referred to that way. Your data model is not like a nib file, where serialized objects are being stored. Instead, what's really happening is you are creating a template from which a new Fetch Request can be instantiate at run time. This is a relatively straight forward process.

One thing that might trip you up is the fact that you don't use `NSMangedObjectContext` to retrieve your Fetch Requests. If you've been using Core Data for a little while, you've probably gotten accustomed to using the context for retrieving, inserting, and editing your data. A Fetch Request, however, is not data. It's part of your data model. Therefore, instead of using the context, you have to use your managed object model (`NSManagedObjectModel`) to retrieve your Fetch Requests. Fortunately, the managed object model is available through an accessor method that gets created automatically for you when you use one of the Core Data project templates. In a Core Data Application project, you can get the managed object model using the `managedObjectModel` method of the application delegate class (***applicationName_AppDelegate***). In a Core Data document-based application, you use the same method, but call it on your document class instead (which inherits the method from `NSPersistentDocument`).

Once you have a reference to your managed object model, it's simply a matter of asking it for your Fetch Request by name. If you aren't using substitution variables, it's very easy. Here's how we would get that first Fetch Request we created:

Getting a Fetch Request from your data model

```
NSManagedObjectModel *model = [self managedObjectModel];
NSFetchRequest *fetch =
  [model fetchRequestTemplateForName:@"olderThanSpouse"];
```

That's all there is to that one. After this call, you're ready to fetch data from the context.

On the other hand, if you have specified substitution variables in your Fetch Request, then you have to build a dictionary containing the substitution variable names as the key values, and the values you wish to replace them with as the corresponding objects. In our case, that means creating an `NSDictionary` with two entries. One with a key value of `@"FROMDATE"` and another with a key value of `@"TODATE"`. The objects to be passed for both of these keys need to be `NSDate` object instances since we are comparing with a date field. If we were comparing a numeric field, we'd use `NSNumber`, and if were comparing a string field, we would use an `NSString`. Here's an example of getting our second Fetch Request out of the data model:

Getting a Fetch Request with substitution variables

```
NSManagedObjectModel *model = [self managedObjectModel];
NSDate *fromDate = [NSCalendarDate
  dateWithNaturalLanguageString:@"1/1/2004"];
NSDate *toDate = [NSCalendarDate
  dateWithNaturalLanguageString:@"12/31/2004"];
NSDictionary *subs = [NSDictionary
  dictionaryWithObjectsAndKeys:fromDate, @"FROMDATE",
  toDate, @"TODATE", nil];
NSFetchRequest *fetch =
  [model fetchRequestFromTemplateWithName:
  @"birthDateBetween" substitutionVariables:subs];
```

Once either of these two chunks of code fires, you will have a pointer to a Fetch Request. It is exactly the same as if you had created it using a format string and a predicate like we did in the previous article, but with less code and without having the format string inside your compiled application.

Conclusion

Storing the definition of your Fetch Requests inside your data model has many advantages. You can use the Predicate Builder to more easily build complex criteria it allows you to move potentially complex format strings out of your compiled code and into your data model. It also allows you to use bind variables to let you alter your Fetch Requests at runtime based on user-provided data.

These tools provide you with the ability to create and use just about any type of Fetch Request you might need right in your data model, and there is very little reason in most situation not to use it. Unfortunately, the fact that the functionality was sort of squirreled away where it wasn't obvious means it is probably being underused, but now that you where it is and how to use it, you can go ahead and get all those yucky format strings and predicates out of your code.

MT

About The Author

Jeff LaMarche wrote his first line of code in Applesoft Basic on a Bell & Howell "Darth Vader" Apple][+ in 1980 and he's owned at least one Apple computer at all times since. In addition to writing, Jeff codes in a variety of languages, with Cocoa Objective-C being, by far, his favorite. Feel free to drop him a line anytime at jeff_lamarche@mac.com.

Lasso Professional Server 8

the fastest, easiest way to tie any database to any website on any platform.

Lasso Professional – the next generation of building data-driven Web sites – is simply the fastest, easiest way to tie **your** databases to the Web. It supports multiple databases simultaneously, eases the sequential migration of databases and it's fully supported on Mac, Windows® and Linux.

4 FREE well commented Lasso solutions to get you started:

Blog: Teaches beginners and advanced users alike how to create an online diary, or web log.

QuickPoll: Add a poll to your Web site with a question and answers, cast a vote or see results.

Collaborate: Enables knowledge sharing among a group of developers and administrators to streamline Web site development from remote locations or geographically dispersed teams.

Message Board: Source code for the Message Board solution is completely open and expandable to suit your needs.

SPECIAL OFFER FOR MACTECH READERS

Try Lasso Professional Server 8 FREE for 30 days.

www.TryLasso.com
lassosales@omnipilot.com
800-678-9958

Then get 60 days of free support with your purchase.

Lasso

OMNIPILOT SOFTWARE

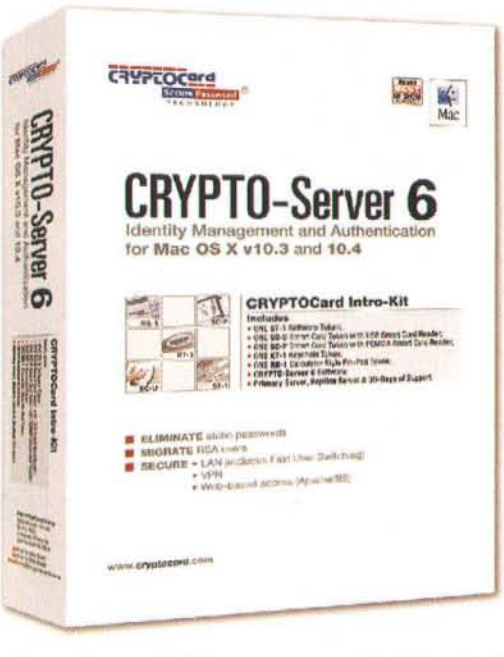
Focus Review *by the MacTech Review Staff*

CRYPTOCARD'S CRYPTO-SERVER 6.3 FOR OS X:

ELIMINATE INSECURE STATIC PASSWORDS

It is no secret that static passwords are the weakest link in the security chain, but until recently, there really was no Mac-specific alternative. That all changed when authentication technology vendor CRYPTOCard released its first two-factor authentication solution for OS X Panther at MacWorld 2004 – winning a MacWorld "Best of Show" award in the process.

CRYPTOCard has now launched a new version of its CRYPTO-Server authentication solution for OS X Tiger. Again, the basic premise of this technology is simple – it replaces inherently weak static passwords with secure two-factor authentication. To log on to a protected network or resource, a user must combine their security PIN (something only they know) with a one-time passcode that is randomly generated by their token for each logon (something only they have).



The new version of CRYPTO-Server does a good job of leveraging Tiger's robust support for smart card environments, but users can also opt for PIN pad tokens, key chain tokens, or software tokens. Each form factor offers unique advantages and characteristics, enabling organizations to tailor their authentication solution according to their own needs. Hardware tokens feature field-replaceable batteries that can be swapped in-service to extend device lifespan indefinitely.

We are pleased to report that our experience testing the new Tiger product was a good one. The CRYPTO-Server package contained everything required to set up the solution, and the instruction manual was clear, accurate, and easy to follow. The product emphasizes ease-of-use and tight integration with Apple's Open Directory LDAP services and as a result, installation was straightforward and relatively painless. There are also features that will simplify implementation in a real-world environment, such as a self-enrolment component called CRYPTO-Deploy, which enables users to remotely assign and activate their hardware tokens via a Web page.

Once the CRYPTO-Server install is completed, a user will need to install the CRYPTO-Console module, an intuitive Graphical User Interface (GUI) which provides the management interface to CRYPTO-Server. CRYPTO-Console enables administrators to manage tokens, users (in non-LDAP deployments), and groups, while also providing server licensing, system configuration, and reporting functions.

The CRYPTO-Console interface is well thought out and easy to navigate, providing administrators with screens for viewing/editing users, tokens, containers, objects, and attributes. Search functions accept regular expressions for ease of use and the GUI architecture is logical and intuitive. Detailed management options are available by highlighting an object, and then Ctrl-clicking it to display drop-down menu items.

The solution appears to be extremely flexible, and can easily be enhanced and expanded with a variety of agents and plug-ins that extend strong two-factor authentication to existing Web, mail, and other security appliance infrastructure nodes. For example, the CRYPTO-Logon for Mac OS X component makes it easy for Mac users attempting to gain secure LAN, Web, or remote access to authenticate themselves by simply inserting their smart card and entering their PIN.

All CRYPTO-Server tokens generate a unique password for every logon attempt, which makes stolen credentials useless to hackers, while simultaneously ensuring Tiger and Panther users do not have to memorize complicated credentials. CRYPTOCard points out that this can significantly reduce the help-desk costs associated with password management while simultaneously eliminating the obvious security risks of "shoulder surfing" and users writing down their passwords.

CRYPTO-Server is also the first solution we have tested that supports two-factor authentication for Apache Web servers via its CRYPTO-Web component. (If you know of others, please let us know!) Using CRYPTO-Web we were able to secure a website, and then authenticate to it with a configured token. CRYPTO-Web should make it a simple process for administrators to secure websites by requiring users to authenticate with their token in order to gain access. Companies can also leverage out-of-the-box interoperability with network entities that provide native RADIUS support.

Unlike CRYPTOCard's original OS X offering, which only provided client side authentication, the latest version of CRYPTO-Server for OS X also provides enterprise-ready functionality like "High Availability" which utilizes real-time multi-master replication functionality to ensure there is no single point of breakdown by switching to a replica server in the event

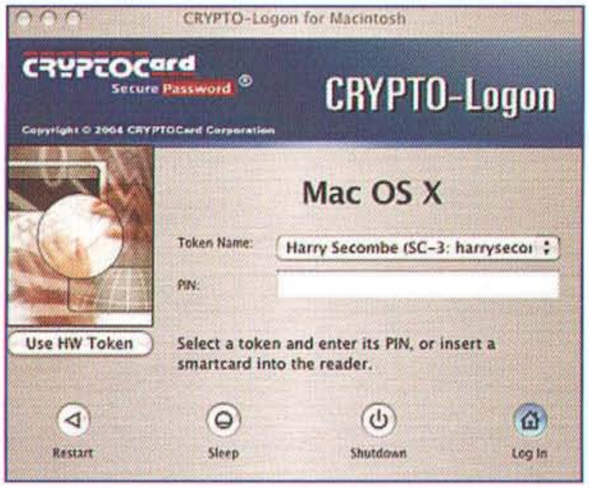
of system failure. This is important as it means that the authentication solution can now meet the security needs of any sized organization.

Another unique feature of CRYPTO-Server is that it offers cross-platform capability. This is important news for the majority of organizations that employ heterogeneous network environments in which any combination of Windows, Linux, or OS X servers can support any client/end-user systems running on any of the three platforms.

Other useful CRYPTO-Server features include RSA migration functionality that enables RSA SecurID DES tokens to be imported into the CRYPTO-Server, and CRYPTO-Kit, a software developer's kit that provides developers with the tools required to integrate CRYPTOCard's technology with existing security applications/systems.

We found CRYPTO-Server for OS X to be very well thought out. Documentation is simple to follow, and the product does a good job of supporting authentication requirements, including a full compliment of token form factors which should make it simple for any sized organization to customize an authentication solution to meet security requirements. The technology makes system configuration simple for administrators, while the familiar ATM-style logon process is easy for users to grasp.

CRYPTOCard was the first authentication vendor to provide real two-factor authentication for the Mac, and we found that the latest version of its technology gives the company a good basis to claim leadership in the OS X authentication marketplace.



Select a token and enter its PIN, or insert a smartcard into the reader.

CRYPTO-Server for OS X is available in a "Five-User Kit," which includes full server software, five tokens of the user's choice, and 30 days support, for $499. This compares favorably with other similar products from other large, well-established vendors. The innovative all-you-need-in-one-box format also makes it simple for an organization to build their security solution as required. CRYPTOCard offers a free trial download of the CRYPTO-Server technology on its website at www.cryptocard.com.

CRYPTOCard Corp.
340 March road
Suite 600
Kanata, Ontario. K2K 2E4
Canada
Phone: : North America 800-307-7042,
International +1-613-599-2441
Fax: +1-613-599-2442
Web: www.cryptocard.com
E-mail: info@cryptocard.com

MT

NOT CURSED, BLESSED!

BUT SOMETIMES, THE KEYS TO SED CAN BE DISGUISED.

This month, we're going to follow up on sed, the powerhouse non-interactive editor, introduced in part 1 in December's issue. Hopefully the holidays of that month didn't take you away from really digging in and putting the lessons into practice. If you did take it all in, you've probably found several uses for it already. This month, we'll visit more sed mnemonics and add a few new tricks.

Bob's Yer Uncle

A few reminders about sed regarding redirection and piping. In part 1, we worked on a single file, viewing the results on screen. There are some things I need to clear up before continuing.

Naturally, you can redirect the output from sed to a file. What I didn't mention last month is a tendency that many first-time sed users need to stay away from. Namely, redirecting to the same file that you're working on. Sure, you've tested your script and you're 100% confident that it's going to Do The Right Thing. So, you do this:

```
sed -f myscript.sed long_text_file.txt >
long_text_file.txt
```

```
Seems perfectly natural, right?  We'll, sorry, you
can't do that.  You can't redirect to the same file
you're altering.  This is not an issue with sed, but
just the way the shell works.  The shell will
truncate the destination file before it executes your
command.  So, while it seems like it might work, it
doesn't.  You need to write to a temporary file
first, and then overwrite the original if you desire
to do so.  The desired effect is achieved like this:
```

```
$ sed -f myscript.sed long_text_file.txt >
tmp_file.txt
$ mv -f tmp_file.txt long_text_file.txt
```

Also, it may not have been clear from the introductory article that sed is happy to work 'on-the-fly' by accepting and pumping out data via a pipe. To get a count of directories in a certain folder, for example, you could do this:

```
ls -lRF /Users/marczak/Documents | grep "^/" | sed
s/:/'\/'/ | sed s/\ /'\\ '/g | wc -l
```

Pipe-to-pipe-to-pipe-to-pipe! However, a sed master will never use two sed statements where one would do. This can also be written as:

```
ls -lRF /Users/marczak/Documents | grep "^/" | sed -e
s/:/'\/'/ -e s/\ /'\\ '/g | wc -l
```

In short, pipe away, redirect carefully, and Bob's yer uncle!

Do it Better

While the information in part 1 of this article is more than enough for very powerful manipulations, you still may run into some limitations. That's why part 1 was the ***introduction***: there's still more!

First, I mentioned that I would give the solution the 'swapping Bill and Michael' problem. The answer, of course, is, "it depends." You really have to be familiar with your source material. For now, I'll show the easy, yet most brute-force way of handling our scenario. Remember, the source text is this:

> Bill and Michael went to the store. Bill needed to buy some butter, eggs and flour. He and Michael were in a hurry to bake a cake for their parent's Anniversary. Once they got home, Bill and Michael realized that they forgot cake icing.

Just like writing any code, you can swap using a temporary variable (sorry – sed doesn't ***quite*** have anything like a bitwise swap!). So, here you go:

```
sed -e 's/Bill/David/g' -e 's/Michael/Bill/g' -e
's/David/Michael/g' short_story.txt
```

In our case, we probably really only want the fully qualified "Bill and Michael", so we can actually do just that:

```
sed 's/Bill and Michael/Michael and Bill/g' short_story.txt
```

However, you may realize, that this does ***not*** take into account line endings. What if our phrase crosses a newline boundary? Ah! That's where ***multi-line*** commands come in.

Multi-line commands give sed the ability to look at more than one line in the pattern space. This gives the sed script-crafter the ability to inject a little logic into the flow. Since our story does not split "Bill and Michael" anywhere, let's look at something that does: "buy some...". If we wanted to change all occurrences of "buy some" to "purchase some", even if it spans lines, we need to coax sed into doing so. Again, the brute-force way is simply this:

```
/buy/ {
N
s/buy *\n*some/purchase some/
}
```

Here, we look for the address "buy" and when found, run a multi-line next ("N") command. This command reads the next line of input and appends it to the current pattern space – still 'separated' by a new line. Like I said, brute-force, and doesn't really scale well. Also, this has the potential of outputting some really long lines. Of course, elegance is just around the corner. A script that gives us use in more general cases could look like this:

```
/buy/ {
N
s/ *\n/ /
s/buy some */purchase milk,\
/
}
```

First, we look for the address "buy", and if we find it, we pull the next line into the pattern space with "N". Then, we can ditch the new line character and replace it with a space. From there we can try to match our patterns. However, even this is a little problematic – ***just a drop***. The example just given works just fine, but let's alter our story and script. Adding a line to the story to make it this:

> Bill and Michael went to the store. Bill needed to buy some butter, eggs and flour. He and Michael were in a hurry to bake a cake for their parent's Anniversary. Once they got home, Bill and Michael realized that they forgot cake icing. It is important that they had not forgotten anything for the special day.

Big Security
for the small business

MacWorld **"Best of Show"** Winner, **CRYPTOCard** introduces **5-User Kit for OS X**, the "All-In-One-Box" Authentication Solution that lets your security grow with you.

CRYPTO-Server 6.3 is the Mac-centric authentication solution for IT infrastructures organized around Mac OS X. Once implemented, your Mac becomes the centralized authentication and token management system for all users, regardless of their computing platform or location.

For Mac Desktops
- Enforced **CRYPTOCard** authentication for desktop access – *Startup, Screen Lock, Sleep*
- Authenticated Fast User Switching
- Internet Connect MSCHAPv2 (Tiger)
- Cisco VPN Client for Mac Integration

Infrastucture Integration
- Open LDAP integration
- Open Directory integration
- Databases (MySQL, Oracle)

Network Integration
- RADIUS-enabled systems – *VPN, Firewall, NAS*
- Web Servers & On-Demand Computing – *Apache, IIS, Citrix*

MACTECH SPECIAL OFFER

5-User Kit *for only* **$249**

* includes 5 tokens, CRYPTO-Server software & 30-day support.

Email **mac@cryptocard.com** and quote **SPMac249-01**.

Promotion can end without notice.

CRYPTOCard® Secure Password TECHNOLOGY
www.cryptocard.com

And changing the script to this:

```
/got/ {
N
s/ *\n/ /
s/got home */returned to their abode\
/
}
```

…will lead to problems. The goal of this script is to change all occurrences of "got home" into "returned to their abode". Run it and see what happens. See the problem? "got" matches "forgotten" on the last line, but makes no substitutions, so the script quits without outputting that line. It's just MIA! What to do? Exempt the last line of the script. Change the "N" to "$!N" – sed recognizes "$" as the last line (not EOL like regexp).

The reality is that you'll find many, many, many examples like this. Depending on your source(s), you may only be able to make a script just *so* general. This goes back to the rule: test, test, test! You can't test your script enough.

Loose Fit

In addition to the main pattern space that sed matches and manipulates, there is also a ***hold buffer***. The commands are pretty self explanatory: 'x' will eXchange the pattern space with the hold buffer. 'h' will copy the current pattern space into the hold buffer, overwriting what was being held previously. 'H' will do the same, but append to the current hold buffer. 'g' gets the contents of the hold buffer and replaces the pattern space. 'G' gets the contents of the hold buffer and appends its contents to the current pattern space.

Before I get into these commands, please remember that sed certainly is a descendent of the phrase TIMTOWTDI – There is more than one way to do it. Many times, there will be multiple solutions to the particular problem you're trying to overcome. Build up one piece at a time, test, and for goodness sake, ***document*** your solution! (Did I mention that sed scripts recognize "#" as a comment?)

So, let's say we want to print the line before and the line after our match, so we can see it in context – like grep's 'A', 'B', and 'C' flags. Here's one way to approach this:

```
sed -n '
'/Anniversary/' !{
# lines that do not match what we're looking for -
save
x
# clear the current pattern buffer with delete
d
}
'/Anniversary/' {
# lines that match
# get the previous line from the hold buffer
x
# print it with p
p
# get the current line back from the hold buffer
x
# print that
p
# get the next line
n
# print it
p
# finally, drop this line into the hold buffer
x
}' short_story.txt
```

BMS

THE LAW OFFICE OF
BRADLEY M. SNIDERMAN

Need help safeguarding your software?

If you're developing software, you need your valuable work protected with trademark and copyright registration, as well as Non Disclosure Agreements.

Then, when you are ready to sell it, you can protect yourself further with a licensing agreement.

I am an attorney practicing in Intellectual Property, Business Formations, Corporate, Commercial and Contract law.

Please give me a call or an e-mail. Reasonable fees.

23679 Calabasas Rd. #558 • Calabasas, CA 91302
PHONE 818-706-0631 **FAX** 818-706-0651 **EMAIL** **brad@sniderman.com**

Going back to our short story, this will look for a line containing 'Anniversary', print the line before it, the line itself, and the line following. Note the use of the "-n" switch passed into sed. This switch tells sed to ***not*** print all output by default. Otherwise, you'll still see all of your input as it filters through sed. Of course, to make all of this more useful, you could drop this right into a shell script, and use $1 for the pattern – this would give you a generic script that will always perform the equivalent of "grep -C 1 pattern file.txt". Just remember: the way I broke this up over several lines is ***very*** bash specific. csh users must use the backslash to tell the interpreter that the line continues on.

Step On

Earlier, I alluded to sed as a programming language by mentioning the classic temp-variable-swap. Well, sed tends to be more full featured than most people realize. You can even implement flow-control! sed features two commands that let you control the logic of your script. 'b' branches to a label. (Reminds me of my favorite Motorola Assembly mnemonic – BRA – BRanch Always). One example, and you'll get it. This script is an alternate to the script just presented that emulates 'grep -C 1':

```
# find our pattern? jump!
/Anniversary/ b printit
# else hold it
h
# jump to end of script
b
:printit
# get previous line from hold buf
x
# print it
p
# get current line back
x
# print it
p
# get next line
n
# print that
p
```

First to note is the ***label***. A label starts a line with a colon, and should contain seven characters or less. While most modern implementations allow a label to be any length – and is actually the POSIX spec – there are still some versions of sed that restrict a label to 7 characters max. With 'b', we just branch – make the jump. Another flow-control command is 't', or, test. Test allows us to branch conditionally. The jump only happens if the previous substitution was successful. Another example is in order. Imagine a file that lists a userclass by number, and it should be by name. Additionally, you must process the file differently for each userclass. Here's a mock script that could handle this:

```
/userclass/{
s/2000/executive/
t excpath
s/1500/management/
```

JobCapture

Automatic Time and Expense Tracking Software

JobCapture is intuitive, user-friendly client-server software that intelligently and automatically tracks working time and activity. Open a document in any application, and it is automatically being timed by JobCapture. This guarantees that *all work activity is recorded.*

- Perfect for ad agencies, design studios, architects, law firms, PR Firms etc.
- Scalable. Flexible.
- Easy to configure.
- Cost effective. Adding additional users is easy and inexpensive.
- Tracks time and expenses.
- Client and Server software is included.
- Easily customized.
- Real-time data delivery.
- Customizable reporting.
- View data numerically or graphically.
- Import/Export data to SQL, mySQL, XML etc.

JobCapture will *increase productivity and your bottom line* by providing accurate records for billing and analysis. *Stop losing money now— Call today* for more information: 973.763.9494. Reps are available 10:00am-5:30pm, EST.

CaptureWorks

www.captureworks.com

```
t manpath
s/1000/staff/
t stpath
# default action
=
b
:excpath
...
b
:manpath
...
b
:stpath
...
# end of script
```

Those examples should get you going with flow control in sed. Remember, you can certainly jump to a label ahead of your test and even get into (basic) recursion!

Harmony

There are a bunch of miscellaneous things that I'd like to point out before we wrap up our conversation about sed. Some of these fall into the "you have to know the rules before you can break them" category, so I waited to present them.

All of the examples I've given, and most that you'll see, use a forward slash ("/") as the delimiter. Amazingly, sed lets you use whatever you like. The delimiter is great if you have a very simple replacement, such as `sed s/one/two/g`. However, if you're replacing file paths, that could get messy. So, use an underscore, if you like: `sed 's_/usr/sbin_/usr/bin_'`. Easier to read, right?

The "=" (equal sign) mnemonic prints out the current line number. So, you can simply line number any text file:

```
sed '=' textfile.txt
```

Or, you can use to just find the lines that the pattern you're looking for lie in:

```
sed -n '/Bill/ =' story.txt
```

You've probably picked these two up by now, but I should make them clear. Quoting: you really only need to quote if you have metacharacters, but it's a good habit to get into. This: `sed s/Bill/Mike/g` is the same as this: `sed 's/Bill/ Mike/g'`. However, try this: `sed s/.*address*\(astring\)\(\1)$1)/` and sed is just going to cough up electrons. The second thing is the use of the "-e" switch. If you only have one command, you can forgo the "-e". If you have multiple commands, you need to add each of them to the list of editing commands with the "-e" switch.

Lastly, a note about newlines, or, EOL. Somewhat confusingly, you match a newline with the standard regexp '\n'. However, to output a newline, you use a literal newline:

```
sed 's/Bill/Bill\
/g' story.txt
```

This will drop a newline after every occurrence of 'Bill' in our sample text.

Cut 'Em Loose Bruce

...and I thought I was going to get to awk this month! Hopefully, this gives you some ideas about sed and its power. I also hope with practice, that you use this power. You really have to see sed as an editor. It just happens to be one that you don't use interactively like vi, emacs or pico. Go forth and edit non-interactively! sed is an indispensable tool for any system administrator's toolchest, and there are plenty of repetitive tasks waiting for you to automate.

MT

About The Author

Ed Marczak owns and operates Radiotope, a technology consulting company. Despite being around the technology block once or twice, he's thankful that there's always something new to look forward to. Something new at http://www.radiotope.com

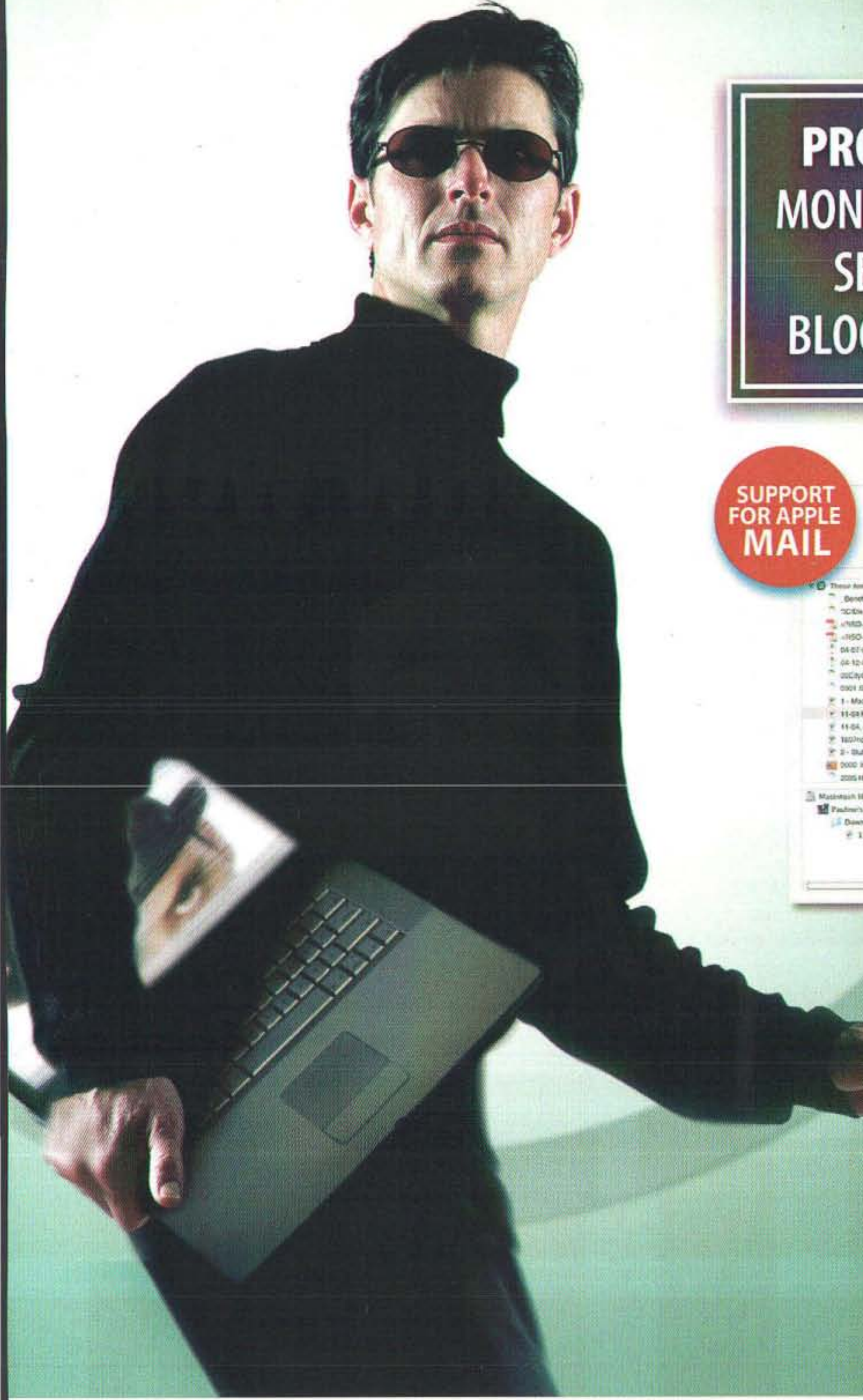
HostedStore™

1. HostedStore provides a **TURNKEY SOLUTION FOR CREATING A WEB STORE**, yet is extensible so developers can use the application as a base for creating customized solutions for their clients.
2. The extensibility of HostedStore provides a structure which allows developers to create modules that add new functionality or alter the built-in features.
3. We are seeking software resellers, hosting providers, developers, and designers that currently offer or plan to offer ecommerce solutions to their clients. Resellers receive a discount off of the published list price of our software and are free to bundle our software with their own products and/or services.

[www.hostedstore.com]

Lasso

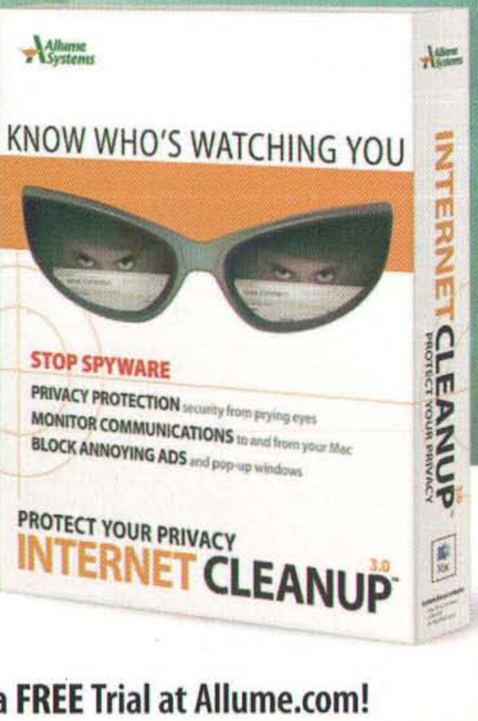
KNOW WHO'S WATCHING YOU.



PROTECT YOUR PRIVACY
MONITOR COMMUNICATIONS
SECURELY ERASE FILES
BLOCK ANNOYING WEB ADS

SUPPORT FOR APPLE MAIL

MailCleaner Search Results
These items were found by MailCleaner
Selected items are targeted for cleaning
Action: Delete
Apply
Size
Kind
Date Modified
Search Results Summary
MailCleaner
Cleaning Action: Move To Trash
Start Over
Finish Cleanup
Show Preview



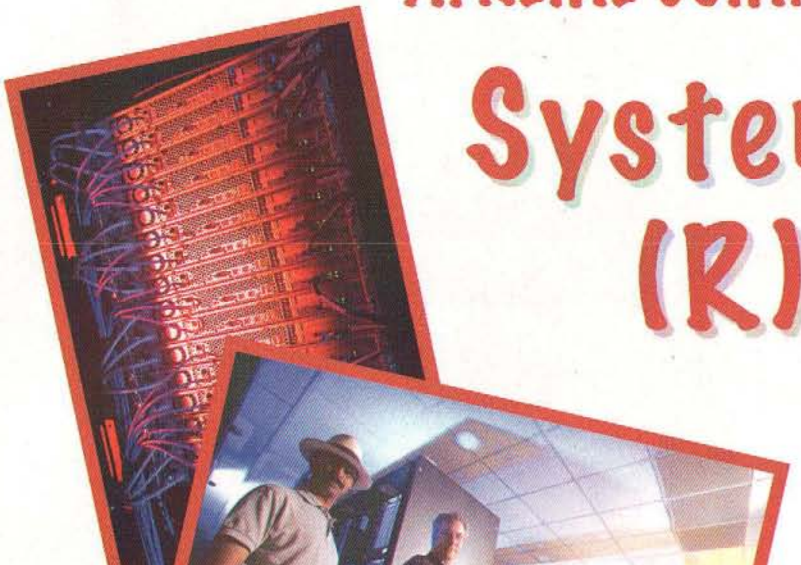
Allume Systems
KNOW WHO'S WATCHING YOU
STOP SPYWARE
PRIVACY PROTECTION security from prying eyes
MONITOR COMMUNICATIONS to and from your Mac
BLOCK ANNOYING ADS and pop-up windows
PROTECT YOUR PRIVACY
INTERNET CLEANUP 3.0

Allume Systems
A Division of Smith Micro Software

Available from your favorite retailers and catalogs. Download a FREE Trial at Allume.com!

One hundred percent 10.4 "Tiger" compatible! Download a FREE trial version of Internet Cleanup at **http://www.allume.com/mac/cleanup/** today!

System-X: (R)evolutions



By Emmanuel Stein

An inside look at the inner-workings of the original OS X-based supercomputer, plans for the final upgrade and the science that drives System-X, as well as, future directions in the world of High Performance Computing.

High Performance Computing (HPC): An introduction

Many of today's HPC Clusters, such as Virginia Tech's System-X, take advantage of open source software and build their infrastructure around commodity hardware solutions in what are typically referred to as "one-off" systems. This type of approach differs dramatically from many of the more turn-key software driven cluster solutions, epitomized by Apple's Workgroup cluster, in which software solutions such as Oracle RAC, gridMathematica and Renderman, just to name a few, drive the hardware solution appropriate for a given application. In the case of these systems, enhanced user experience and facilitated deployment and management represent the driving forces. However, such solutions compromise efficiency for ease of use and therefore tend to take the performance out of HPC. Among today's real HPC solutions, a premium is placed on getting the most out of a given hardware architecture. In the case of System-X, as well as, similar clusters that have followed (e.g. Turing and COLSA HPC clusters), the PowerPC 970 processor, with its two double precision floating units and the availability of low-latency fabric such as Myrinet and InfiniBand, served as the prime movers behind the choice of platform. The majority of today's top HPC supercomputers take advantage of these low-latency solutions to realize their blindingly fast parallel computations. This, in turn, enables a cluster to take full benefit of the processing power of each individual node.There are a number of great vendors providing low-latency technology including Small Tree Solutions and Mellanox.

To take advantage of an HPC environment, Message Passing Interface (MPI) is employed to realize parallelization across a targeted set of cluster nodes. MPI is a low-level, but standardized method of programmatically enabling communication among disparate nodes. Implemented as libraries in C and subroutines in FORTRAN, MPI was developed as a portable, low-level interface for enabling parallelism within a multi-processor or multi-node environment. Although, other mechanisms for breaking down computational problems across multiple processors or nodes may be accomplished using approaches such as OpenMP, MPI remains the interface of choice for HPC applications due to its flexibility, raw performance and scalability. For example, Verari Systems Software makes the commercial MPI/Pro 2.1 software, considered to be a high-performance, scalable implementation of the MPI-2 standard; and Dauger Research makes Pooch, a graphical MPI solution.

System-X: The Race to Build Academia's Fastest Supercomputer

At 1pm, on June 23, 2003, Jason Lockhart watched on as Apple announced the new PowerMac G5. In that instant the frustration of months of dead end talks with potential vendors, for the acquisition of an HPC cluster capable of 10 teraflops, vanished as the future platform of the then unnamed System-X materialized as if out of thin air. A week later Dr. Srinidhi Varadarajan, who had submitted a proposal to the National Science Foundation (NSF) for a Major

Research Instrumentation grant, was flown to Apple headquarters to discuss the feasibility of constructing a 1,100 node HPC supercomputer using Apple's yet to be released line of PowerPC 970-based systems. Varadarajan not only convinced Apple that Virginia Tech should be given the opportunity to build the high profile HPC cluster, but also managed to secure the first 1,100 production units. Although, not a Mac user at the time, Varadarajan took only 3 days to realize that OS X and the PowerMac G5 represented the ideal platform for building his ambitious supercomputing project.

Grace Under Pressure:

Following the choice of the G5 platform came the monumental task of adapting and extending core software components such as MPI, BLAS, InfiniBand drivers, and the custom high performance memory manager to run optimally on the 2,200 CPU cluster. In a whirlwind of prodigious programming, Varadarajan, 2 Mellanox engineers and a handful of graduate students accomplished what would have taken an army of dedicated developers many months to complete.

Among the most crucial tasks, was to adapt existing variants of MPI to run on such a large-scale cluster and, moreover, to do so in a manner that was both robust and that took full advantage of the low latency InfiniBand fabric. Although, MPI variants had been ported to OS X previously, none offered the ability to scale much past a 128-node cluster, much less a 2,200 CPU system. There was also the challenge to adapt the existing MPI stack to make efficient use of the InfiniBand fabric. In fact, the only available MPI implementation that did support InfiniBand was only available in beta form and had not been designed, much less optimized for either OS X or a cluster on the scale of System-X.

In collaboration with Dr. Panda, of Ohio State University, the author of the MVAPICH (MPI for Verbs API with InfiniBand support), Varadarajan worked tirelessly to sort through the MVAPICH code base, optimizing or rewriting substantial portions of the code. In this endeavor, virtually no element of the code was left untouched, since the ultimate performance and ranking of System-X would depend on the successful implementation of this critical piece of the puzzle.

At the same time, two Mellanox engineers aided by then graduate student Michael Heffner, worked around the clock for several weeks to develop the raw InfiniBand drivers for use with System-X. These raw drivers were integrated with the newly ported MPI stack using a kernel bypass to communicate directly with the InfiniBand Host Channel Adapter (HCA).

During this period of frantic development, Varadarajan, with extensive help from Dr. Goto, of the Texas Advanced Computing Center, worked to further tune and optimize Gotos' BLAS (Basic Linear Algebra Sub-routines) for use with his memory management system. GotoBLAS is the fastest available BLAS implementation and proved instrumental in obtaining increased performance of matrix operations—crucial for high scores on the LINPACK benchmark (http://www.netlib.org/benchmark/hpl/) used by the Top500 organization to rank supercomputers.

What resulted from these intense efforts was what principal designer Varadarajan termed a "pleasant surprise," with an amazing top 3 showing in the supercomputing challenge. This accomplishment is underscored by the fact that System-X cost only $5.2 Million and was put together in roughly 3 weeks! To put this in perspective, the top supercomputer that year, the custom designed Japanese Earth Simulator (5,104 processors), which outperformed System-X by a factor of three, cost $350 Million and was several years in the making.

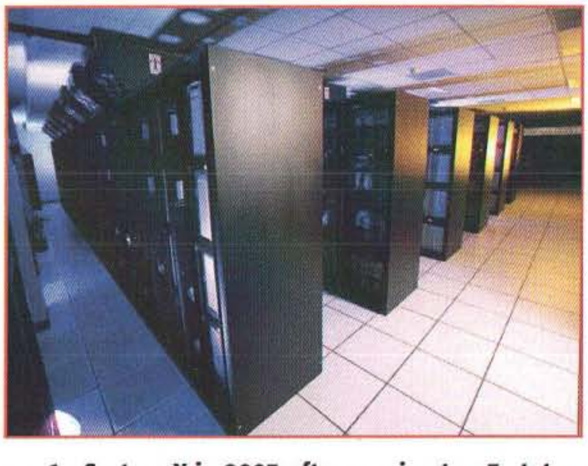


Figure 1. System-X in 2003 after earning top 3 status on the Supercomputing Top500 with 10.28 TF

System-X Version 2.0

In 2004 System-X was upgraded from 1,100 PowerMac G5 towers to Xserve G5s running dual 2.3 GHz processors, up from the previous towers running dual 2 GHz G5s. The improved throughput enabled by the Xserve I/O subsystem, coupled with the increased clock speed of the processors, afforded the system a significant speedup, from 10.28 to 12.25 teraflops (peak performance 17.5 teraflops). With the addition of ECC memory the system moved from, what was, in essence, a benchmarking and proof-of-concept system, to a formidable production HPC supercomputer capable of precise and massively parallel scientific computations. The addition of ECC RAM proved essential to the success of subsequent research since, over 1100 nodes, a single bit flip, caused by a solar flare, could significantly corrupt the precision of the calculated data.

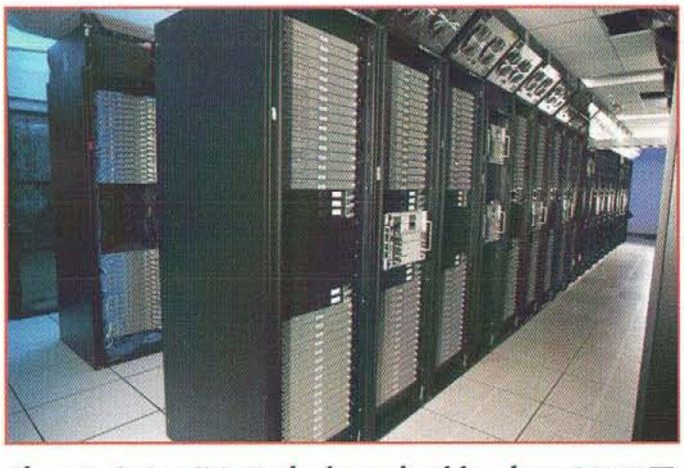


Figure 2. System-X: In Production and cruising along at 12.25 TF

System-X: The Makings of an HPC Speed Demon

Compute Nodes:

1,100 Dual 2.3Ghz Xserve Cluster Nodes configured with 4GB of ECC DDR400 RAM, 80GB hard drives, Gigabit Ethernet and Mellanox Cougar 4x InfiniBand Host Channel Adapters (HCA).

Compile Nodes:

3 Dual 2.3Ghz Xserve nodes with 4GB of ECC DDR400 RAM, 3 250GB hard drives and Gigabit Ethernet.

Networking:

For primary low latency fabric, 4 SilverStorm Technologies 9120 InfiniBand core switches, supporting 4x InfiniBand (10Gbs bidirectional port speed) with 11 leaf modules and 3 spine modules, as well as, 64 SilverStorm Technologies 9024 InfiniBand leaf switches using 4x InfiniBand (10Gbs bidirectional port speed) with 24 InfiniBand ports per leaf switch. Secondary communications provided by 6 Cisco Systems 240-port 4506 Ethernet switches.

Storage:

Xserve RAID, configured as a RAID 50 array, storing 2.7TB of data and available as an NFS server with aggregate write bandwidth of 90-100MB/sec. System-X users employ this as a temporary storage area with results sets offloaded to more permanent storage areas as needed.

Cooling:

Custom Liebert Extreme Density cooling system, in a chilled water loop configuration and fed off of two 125 ton Carrier water chillers that supply about 3 million BTUs of cooling capacity. Liebert XDP systems with a R-134A refrigerant loop supplied to rack mounted liquid-to-air heat exchangers.

Software:

Apple OS X 10.3.9 (currently migrating to 10.4.x), MVAPICH for message passing, Torque (OpenPBS) for queue management, Moab (Maui) for job scheduling, Ganglia for system monitoring, as well as, IBM XL Fortran, IBM XLC and GCC 3.3 compilers. For example, Verari Systems Software makes the commercial MPI/Pro 2.1 software, considered to be a high-performance, scalable implementation of the MPI-2 standard; and Dauger Research makes Pooch, a graphical MPI solution.

System-X: Terminal Velocity

Although System-X runs a relatively unmodified version of OS X, the GUI is absent from compute nodes and job submission is done via the simple yet powerful terminal interface. What follows is a cockpit view of System-X.

The `pbstop` command serves as a monitor of node activity across the cluster.

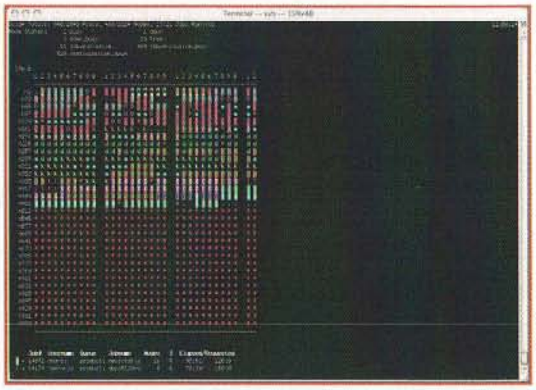
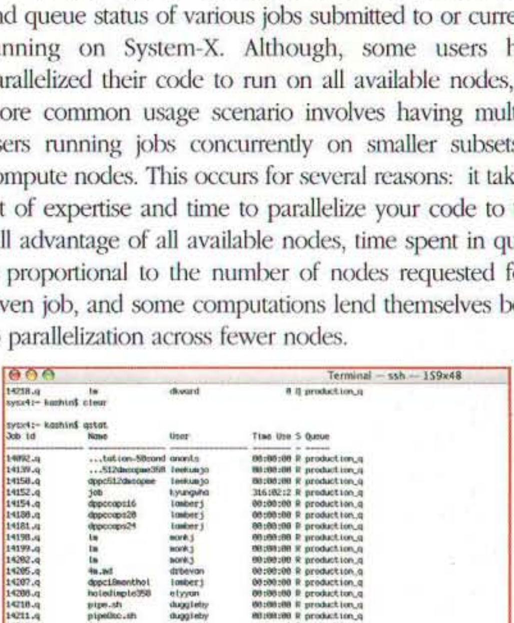


Figure 3. pbstop command

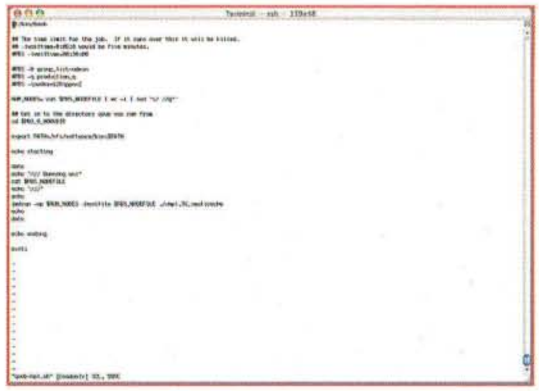
With the `qstat` command, users can see the activity and queue status of various jobs submitted to or currently running on System-X. Although, some users have parallelized their code to run on all available nodes, the more common usage scenario involves having multiple users running jobs concurrently on smaller subsets of compute nodes. This occurs for several reasons: it takes a lot of expertise and time to parallelize your code to take full advantage of all available nodes, time spent in queue is proportional to the number of nodes requested for a given job, and some computations lend themselves better to parallelization across fewer nodes.



```
14218.q          lm               dkward                   0 Q production_q
sysx4:~ kashin$ clear

sysx4:~ kashin$ qstat
Job id           Name             User             Time Use S Queue
---------------- ---------------- ---------------- -------- - -----
14092.q          ...tution-50cond ananls           00:00:00 R production_q
14139.q          ...512dmcopme350 leekumjo         00:00:00 R production_q
14150.q          dppc512dmcopme   leekumjo         00:00:00 R production_q
14152.q          job              kyungwha         316:02:2 R production_q
14154.q          dppccops16       lomberj          00:00:00 R production_q
14180.q          dppccops20       lomberj          00:00:00 R production_q
14181.q          dppccops24       lomberj          00:00:00 R production_q
14198.q          lm               monkj            00:00:00 R production_q
14199.q          lm               monkj            00:00:00 R production_q
14202.q          lm               monkj            00:00:00 R production_q
14205.q          4m.md            drbevan          00:00:00 R production_q
14207.q          dppc18menthol    lomberj          00:00:00 R production_q
14208.q          holedimple350    elyyan           00:00:00 R production_q
14210.q          pipe.sh          duggleby         00:00:00 R production_q
14211.q          pipeDsc.sh       duggleby         00:00:00 R production_q
14212.q          job              kyungwha                0 Q production_q
14213.q          dimplesq200      rozati           00:00:00 R production_q
14214.q          flow.sh          duggleby         00:00:00 R production_q
14216.q          lm               dkward                  0 Q production_q
14217.q          flow-low-rot     ananls                  0 Q production_q
14218.q          lm               dkward                  0 Q production_q
sysx4:~ kashin$ ▊
```

Figure 4. qstat command

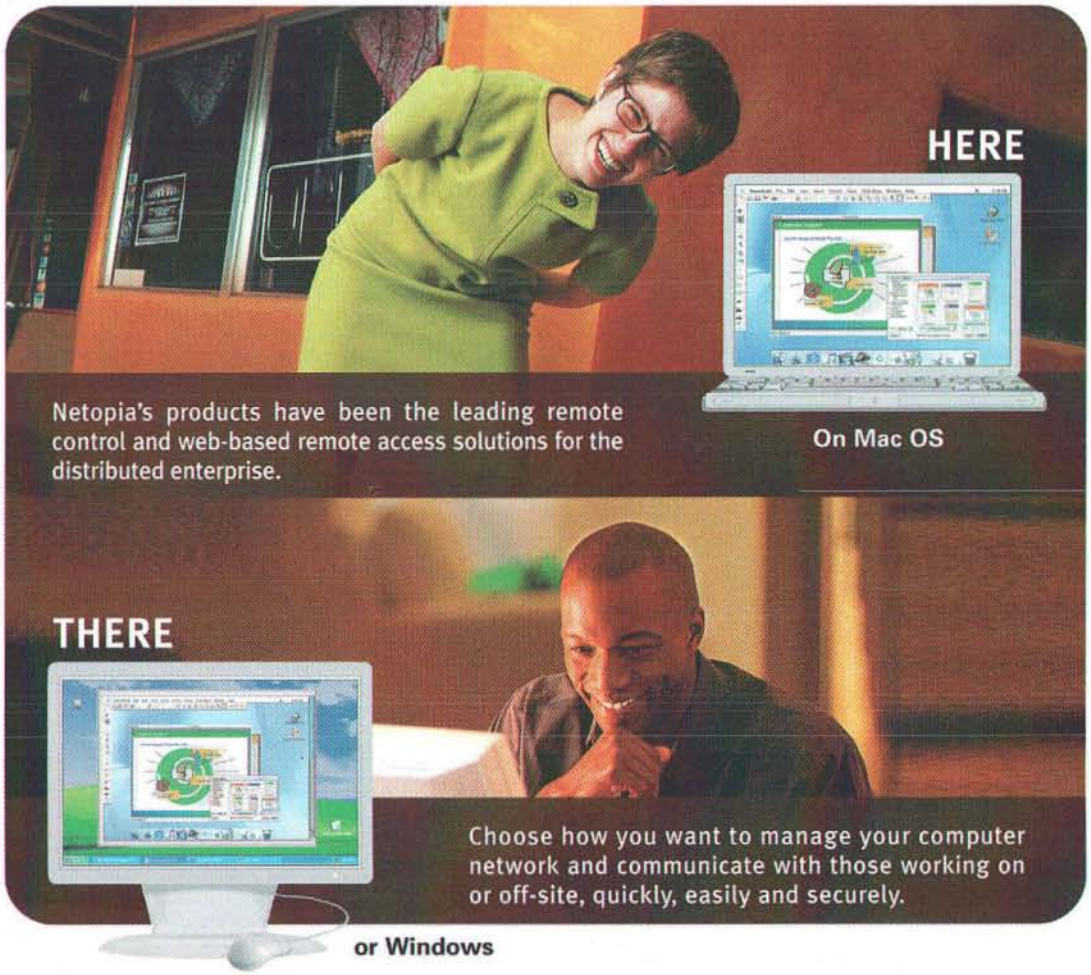
Below is a generic job submission script for System-X. Interestingly enough, it is also employed as a simple benchmarking routine to test system performance and troubleshoot possible issues across system nodes.



```
#!/bin/bash

## The time limit for the job. If it runs over this it will be killed.
## -l walltime=0:05:00 would be five minutes.
#PBS -l walltime=00:30:00

#PBS -N [illegible]
#PBS -q production_q
#PBS -l [illegible]

NUM_NODES=`cat $PBS_NODEFILE | wc -l | sed "s/ //g"`

## Get in to the directory that you ran from
cd $PBS_O_WORKDIR

export PATH=[illegible]

echo starting

date
echo "\\\ Running on:"
cat $PBS_NODEFILE
echo "\\\"
echo
mpirun -np $NUM_NODES -machinefile $PBS_NODEFILE [illegible]
echo
date

echo ending

exit
~
"qsub-hpl.sh" [readonly] 33L, [illegible]
```

Figure 5. qsub.sh

Stay In Control Wherever You Go.



Netopia's products have been the leading remote control and web-based remote access solutions for the distributed enterprise.

Choose how you want to manage your computer network and communicate with those working on or off-site, quickly, easily and securely.

Timbuktu® Pro

Whether you're at home or at work, using a Mac OS or Windows platform, Timbuktu Pro allows you to operate distant computers as if you were sitting in front of them. Transfer files and folders quickly and easily, and communicate by instant message, text chat, or voice intercom.

eCare

eCare's web-based remote access capabilities enable any user to securely place their virtual eyes and fingers directly on the remote client's desktop for collaboration, problem resolution, and improved customer satisfaction. Using only a web browser, eCare's secure and fast screen sharing, file transfer, live chat, URL push and live session recording make it the ideal solution for the conference room, classroom, or help desk – live, on demand.

Download an evaluation or buy it online: www.netopia.com
Call us at 1-800-485-5741

timbuktu® pro • eCare

netopia.

She Blinded Me With Science

System-X currently plays host to numerous researchers around the world, thereby setting the stage for increasingly groundbreaking research. In this section we will cover some of the most popular applications available on System-X and feature some of the stunning scientific developments being made with the advantage of 12.25+ teraflops of raw number crunching power.

A Community of Codes

System-X was designed to offer a generalized, rather than application specific (e.g. IBM's Blue Gene Machines) platform for scientific research. To capitalize on the performance capabilities of the machines, prospective researchers must become master parallel programmers in addition to competent scientists wuthin their respective disciplines. This rather daunting requirement is somewhat offset by the widespread availability of discipline-specific community code that is used as a starting point for the computational decomposition of scientific questions. Although, standards like MPI have made code portability among these systems more of a reality, each individual HPC cluster represents a unique computing environment and thus no two HPC clusters are identical. As a result, considerable effort is required in optimizing code for parallel runs specific to a given HPC environment.

What follows represents some of the community derived software tools that are available as part of the System-X:

- AMBER (Assisted Model Building with Energy Refinement): A molecular dynamic simulation package based on a molecular dynamics force field equation developed by Peter Kollman ***et al.*** at the University of California. The simulation of protein folding is among the variety of applications for which AMBER is used.

- ARPREC (C++/Fortran-90 Arbitrary Precision Package): Essentially a C++ Library with both C++ and Fortran-90 translation modules, ARPREC is a mathematical toolkit that enables researchers to conduct interactive and high-precision arithmetic computations up to the level of 10-million digits. This computational environment supports high-precision real, integer and complex datatypes and may be applied to all basic arithmetic operations, most transcendental and combinatorial functions, as well as, high-precision quadrature and summation of series.

- ARPS (Advanced Regional Prediction System): This advanced weather modeling tool, developed at the Center for Analysis and Prediction of Storms (CAPS), is an atmospheric modeling and prediction system, which affords researchers real-time data analysis and assimilation.

- CHARMM (Chemistry at HARvard Macromolecular Mechanics): In the same vein as AMBER, CHARMM is both a force field and a simulation package that allows scientists to apply the force field algorithms to problems in molecular dynamics.

- FASTEST (Flow Analysis Solving Transport Equations with Simulated Turbulence): Applied in the study of fluid dynamics, FASTEST is used to solve three-dimensional flow problems. The set of associated equations include several turbulence models, which feature free topology, implicit time stepping, and that are parallelized and multi-grid enabled.

- GAMESS (General Atomic and Molecular Electronic Structure System): Used by researchers in quantum chemistry, GAMESS is specifically designed software package for performing computational chemistry calculations of the Hartree-Fock, density functional theory and configuration interaction types, among others, within the domain of advanced electronic structure methods.

- Global Arrays: A shared memory programming interface for use in distributed memory environments such as computational clusters, this toolkit represents a departure from other shared memory interfaces inasmuch as it is fully compatible with Message Passing Interface (MPI) and, furthermore, allows programmers to combine message passing and shared memory schemes within the context of a single application. Global Arrays also goes beyond MPI compatibility by allowing users to exploit existing message-passing libraries for facilitated development of highly parallelized code.

- LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator): This molecular dynamics simulator is designed for use in both parallel and single processor environments and may be used to build molecular systems, automatically assign force field coefficients and perform complex molecular dynamics simulations analyses and visualizations. LAMMPS may be applied to atomic, polymetric, biological, metallic, granular, and hybrid systems. Written in portable C++, LAMMPS is highly extensible and is capable of distributed memory message passing for optimal performance in HPC environments.

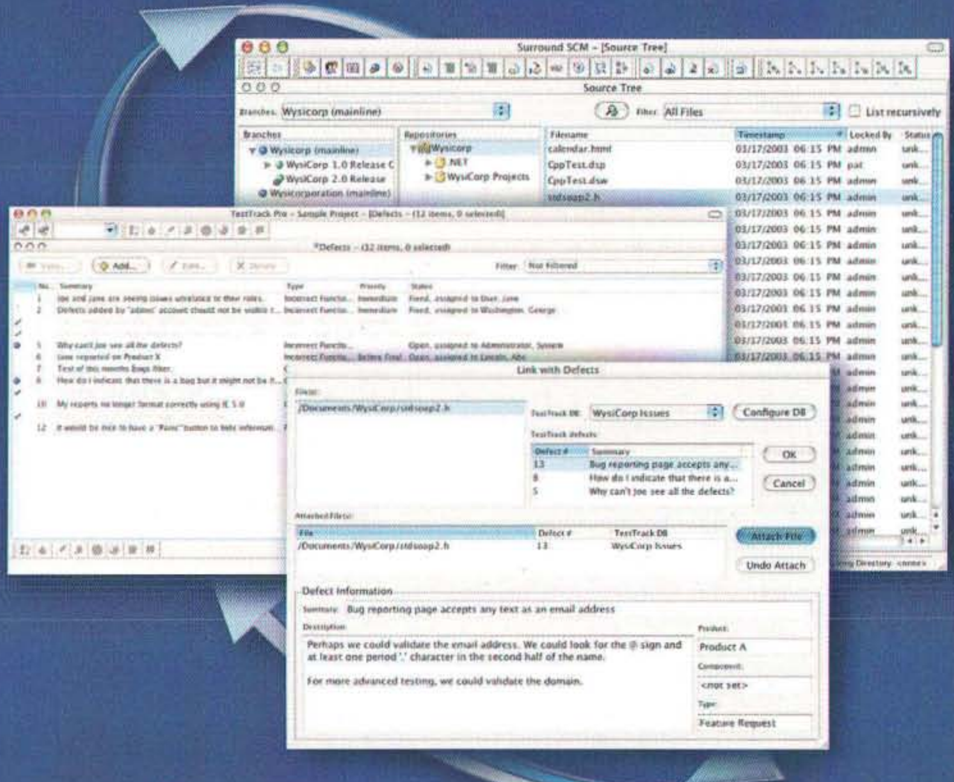
- METIS (Family of Multi-Level Partitioning Algorithms) and parMETIS (Parallel METIS): METIS and its parallelized, MPI aware variant parMETIS represent a series of applications for partitioning unstructured graphs/hypergraphs and are used in the computation of fill-reducing orderings of spare matrices. This family of programs is applied in the fields of numerical computation, VSLI, geographical information systems, operation research, bioinformatics and knowledge discovery.

- NWChem: An HPC-enabled computational chemistry package used in the study of molecular mechanics and molecular dynamics using Hartree-Fock, Post-Hartree-Fock, and density functional theory methods. The NWChem software was developed at the Pacific Northwest National Laboratory by the Molecular Sciences Software group within the context of the Theory, Modeling & Simulation program and under the purview of the Environmental Molecular Sciences Laboratory.

- PETSc (Portable, Extensible Toolkit for Scientific Computation): A suite of data structures and routines used in the parallelized computation of both linear and nonlinear equations and available in C, C++, Fortran and Python.

Complete Source Control and Defect Management for Mac OS X

Seapine Software™
changing the world of software development



Surround SCM - [Source Tree]

Source Tree

Branches: Wysicorp (mainline) | Filter: All Files | List recursively

Branches	Repositories	Filename
Wysicorp (mainline)	Wysicorp	calendar.html
WysiCorp 1.0 Release C	.NET	CppTest.dsp
WysiCorp 2.0 Release	WysiCorp Projects	CppTest.dsw
Wysicorporation (mainline)		stdsoap2.h

Timestamp	Locked By	Status
03/17/2003 06:15 PM	admin	unk...
03/17/2003 06:15 PM	pat	unk...
03/17/2003 06:15 PM	admin	unk...
03/17/2003 06:15 PM	admin	unk...
03/17/2003 06:15 PM	admin	unk...
03/17/2003 06:15 PM	admin	unk...
03/17/2003 06:15 PM	admin	unk...
03/17/2003 06:15 PM	admin	unk...
03/17/2003 06:15 PM	admin	unk...
03/17/2003 06:15 PM	admin	unk...
03/17/2003 06:15 PM	admin	unk...

TestTrack Pro - Sample Project - [Defects - (12 items, 0 selected)]

*Defects - (12 items, 0 selected)

Filter: Not Filtered

No.	Summary	Type	Priority	Status
1	Joe and Jane are seeing issues unrelated to their roles.	Incorrect Functio...	Immediate	Fixed, assigned to Doe, Jane
2	Defects added by "admin" account should not be visible t...	Incorrect Functio...	Immediate	Fixed, assigned to Washington, George
5	Why can't Joe see all the defects?	Incorrect Functio...		Open, assigned to Administrator, System
6	Issue reported on Product X	Incorrect Functio...	Before Final	Open, assigned to Lincoln, Abe
7	Test of this months Bugs filter.			
8	How do I indicate that there is a bug but it might not be it...			
10	My reports no longer format correctly using IE 5.0			
12	It would be nice to have a "Paste" button to hide informati...			

Link with Defects

Files: /Documents/WysiCorp/stdsoap2.h

TestTrack DB: WysiCorp Issues

TestTrack defects:

Defect #	Summary
13	Bug reporting page accepts any...
8	How do I indicate that there is a...
5	Why can't Joe see all the defects?

Attached File(s):

File	Defect #	TestTrack DB
/Documents/WysiCorp/stdsoap2.h	13	WysiCorp Issues

Defect Information

Summary: Bug reporting page accepts any text as an email address

Description: Perhaps we could validate the email address. We could look for the @ sign and at least one period '.' character in the second half of the name.

For more advanced testing, we could validate the domain.

Product: Product A

Component: <not set>

Type: Feature Request

Effective source code control and defect tracking require powerful, flexible, and easy-to-use tools—Surround SCM and TestTrack Pro

- Complete source code control with private workspaces, automatic merging, role-based security, and more
- Comprehensive defect management — track bug reports and change requests, define workflow, customize fields
- **New!** Full Unicode support maintains international characters across various languages, ensuring data is not lost or misinterpreted
- Advanced branching simplifies managing multiple versions of your products
- Fast and secure remote access to your source files and defects — work from anywhere
- Scalable and reliable cross-platform, client/server solutions support Mac OS X, Windows, Linux, and Solaris
- Exchange data using XML and ODBC, extend and automate with SOAP support
- Licenses priced to fit your budget

Seapine Software Product Lifecycle Management
Award winning, easy-to-use software development tools

Seapine **Surround SCM**

Seapine **TestTrack** *PRO*

software development 13th annual product excellence award JOLT

Download Surround SCM and TestTrack Pro at www.seapine.com/mac or call 1-888-683-6456

- ScaLAPACK (Scalable Linear Algebra PACKage): Used in the solution of dense linear systems and for the computation of eigenvalues for dense matrices, these block-partition based algorithms serve to curtail the frequency of data movement between levels within memory hierarchies. Here, memory hierarchies can apply to the off-processor memory of other processors, hierarchies of registers, cache and local memory on individual processors.

- Unified Parallel C: A set of C programming extensions used to provide a framework for programming in the context of shared, as well as, distributed memory environments

- VASP (Vienna Ab-initio Simulation Package: A Fortran-90 application package used in the field of ab-initio quantum-mechanical molecular dynamics and for the purposes of calculating forces and stresses involved in relaxing atoms into their respective instantaneous groundstates.

- VecLib: (BLAS, LAPACK, FFT, DSP) An Apple designed framework for abstracting intensive mathematical processing in a manner optimized for the altivec engine and specific to the G5 processor. For more details see http://developer.apple.com/technotes/tn/tn2086.html,

- WRF (Weather Research and Forecasting Model: This mesoscale numerical weather prediction system is used for operational forecasting and atmospheric research and is useful for climate modeling across meters to thousands of kilometers.

A Gallery of Computations

To highlight some of the real-world science coming out of System-X, the following is a sampling of research visualizations, with accompanying quotes from the principal investigators. QuickTime renders of the screenshots below, including additional material, are available at http://people.cs.vt.edu/~ribbens/tcf/SC05/, courtesy of Cal Ribbens, Ph.D.

"Dr. Onufriev's group is using System-X to perform a series of molecular dynamics simulations aimed at gaining insight into (and ultimately proposing a mechanism for) the unusual flexibility of short DNA fragments. This new phenomenon was recently discovered in experiments that challenge the conventional picture of the DNA molecule, traditionally thought to behave more or less like a rigid rod at the biologically important length scales of up to tens of nanometers. The emerging picture of a much more flexible DNA may change our understanding of how DNA interacts with other biomolecules, such as proteins. An ability to correctly describe these interactions is of fundamental importance to molecular biology and medicine. In agreement with experiments, computer simulations reveal large-scale fluctuations of the DNA fragment, and most importantly provide full atomic details of the structural changes upon bending. Early results indicate that no unraveling of the famous double-helical structure is required for substantial bending, helping to zoom in on the possible mechanism explaining this newly discovered phenomenon." - Alexey Onufriev

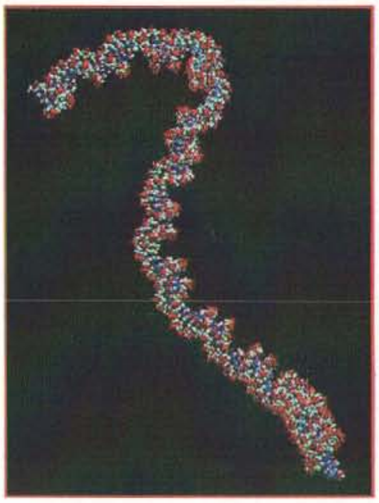


Figure 6. Insights into the Mechanisms of DNA Flexibility (Alexey Onufriev)

"Phospholipid bilayers are cell membranes which play an essential role in protecting the cell and regulating biological activity between the extracellular and intracellular domain. The biophysics of cell membranes is of great importance in understanding biological phenomena, ranging from drug interaction to cancer treatment to bio-preservation. Conventional laboratory experiments are not always capable of probing molecular interactions and distinguishing among the many processes occurring at the molecular level. Professor Sum and his research group (www.che.vt.edu/Sum) are applying and developing advanced molecular modeling methods to study phospholipids bilayers to understand their structure, dynamics, and interactions with different solute molecules. One project models the interactions and diffusion of cryoprotectants with the phospholipids bilayer to obtain insight into preservation mechanisms of biological systems. In another project, Sum and colleagues are probing the activation mechanism of the sensory system in response to specific compounds. Even though these systems only encompass a few nanometers of a model cell membrane, they contain a large number of molecules. System X has enabled Professor Sum to study these systems in much greater detail by allowing extensive simulations for larger systems (order of 100,000 atoms) and for longer periods of time (order of 10-100 ns). The simulations on System X allow for the development of long-time dynamics, which is critical to understanding the phospholipids bilayers (long-relaxation times) and the diffusion process." - Amadeu Sum

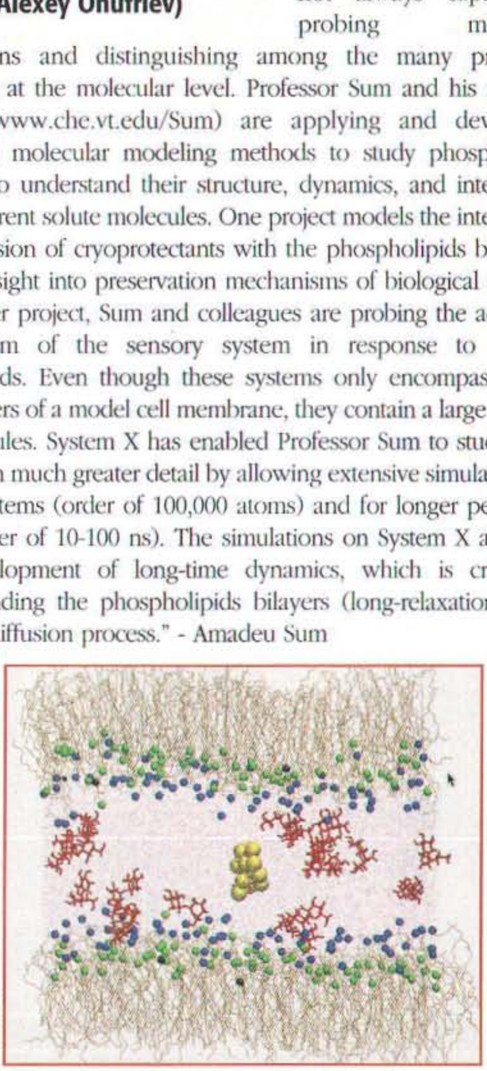
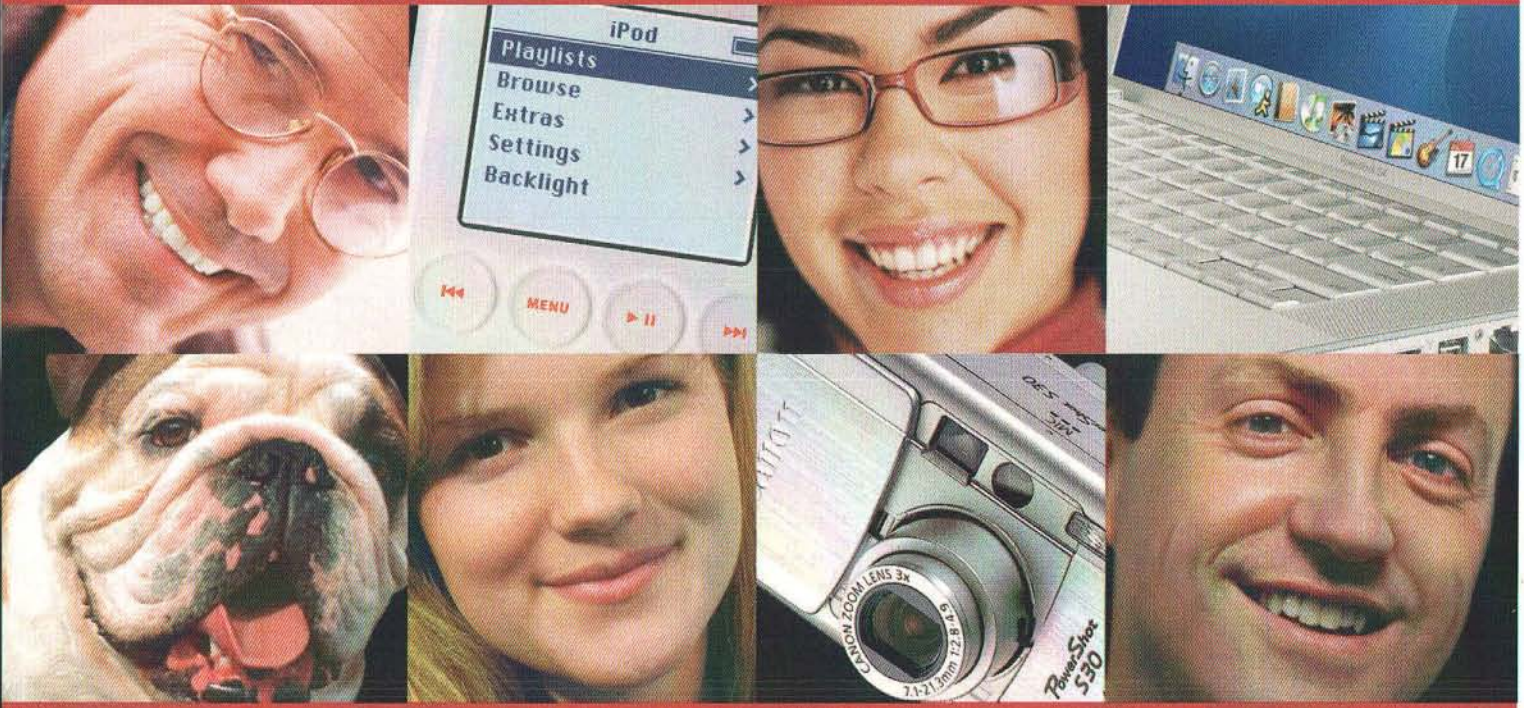
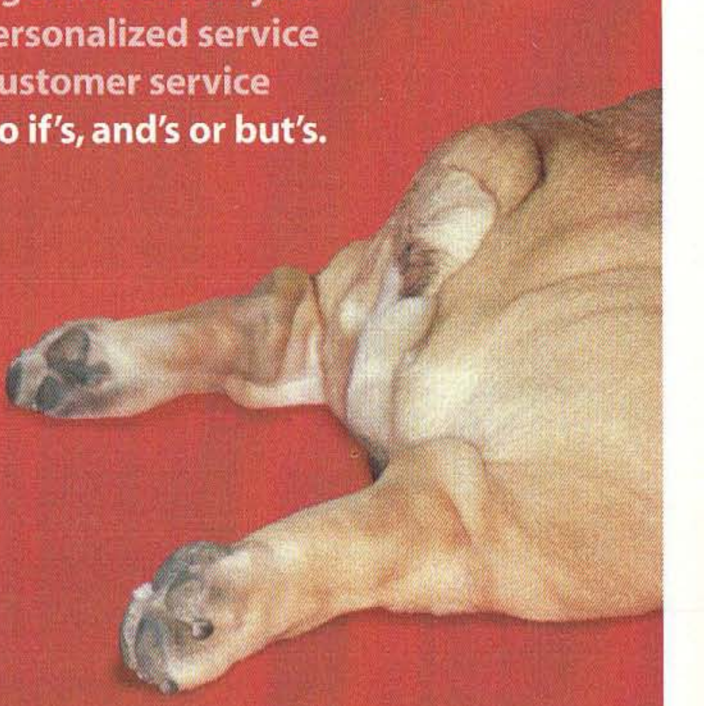


Figure 7. Biophysics of Phospholipid Bilayers (Amadeu Sum)

At Small Dog Electronics, happy customers are our highest priority.



When you shop at Small Dog Electronics, you get more than just great selection and low prices; you also get personalized service from genuine Apple Professionals who take customer service very seriously. And that's a promise... **no if's, and's or but's.**



Small Dog Electronics
www.smalldog.com

Apple Specialist

1-800-511-MACS
A socially responsible business since 1996

System-X Version 3.0

Virginia Tech is in the process of upgrading System-X to take advantage of technological advances in both software and hardware and thereby increase the performance of this computational titan. Specific upgrades include a general system update from OS 10.3.9 to OS 10.4, which promises to offer benefits in terms of better memory management and, most significantly, the ability to use 64-bit address space to dramatically increase the resolution of experimental results sets. Further enhancements will include updated InfiniBand drivers, as well as, the deployment of newer versions of the Mellanox InfiniBand HCAs used for inter-node communication. Beyond these enhancements, each node will be upgraded from 4 to 8 Gigabytes of memory, thus doubling the available memory pool for distributed computation. In additional, there are plans to update the MPI stack and deploy 64-Bit compilers to take advantage of the addressing space in Tiger. Once these updates are applied, System-X will be considered stable and no further enhancements, beyond minor software updates, are envisaged. The goal with this final revision of System-X is to offer a robust production HPC platform that will service the scientific community for years to come, with longevity being a primary aspiration.

Figure 8. System-X today running at full bore and ready for Tiger!

Beyond System-X

System-X is only the beginning for Virginia Tech, which has opened the Center for High-End Computing Systems (CHECS). Under the direction of Dr. Varadarajan, the center's mission is to tackle the various challenges associated with developing the next generation of HPC clusters. The staff are currently working to train future computational scientists in the art of building HPC-enabling and enhancing technologies. One of the primary goals is a convergence of hitherto disparate CS disciplines such as processor and memory architectures, operating systems, runtime environments, communications subsystems, fault-tolerance, scheduling and load-balancing, power aware systems and algorithms and associated programming models. According to Varadarajan, the aim is to architect and implement "computing systems and environments that can efficiently and usably span the scales from department-sized machines to national-scale resources." One example of this is the National Lambda Rail (NLR), an Ethernet based optical network, which serves to connect research communities and associated computing resources across the country. Although not yet in widespread use, the NLR promises to advance research in the area of next generation networking technologies and its derivatives may well overtake today's traditional Internet backbone. CHECS will also serve as a test-bed for both theoretical and practical applications within the area of HPC and will work to develop successors to System-X, as well as, collaborate with scientific communities across the world to create much needed software environments for future HPC systems.

Credits

This article would not have been possible without the generous participation of the following individuals.

Srinidhi Varadarajan, Associate Professor, Computer Science Department
Director, Center for High-End Computing Systems
http://www.checs.eng.vt.edu

Cal Ribbens
Associate Professor, Associate Department Head, Computer Science Department
Director, Laboratory for Advance Scientific Computing and Applications (LASCA)
http://research.cs.vt.edu/lasca/

Jason Lockhart, Director of High-Performance Computing and Technology Innovation, College of Engineering
http://www.eng.vt.edu

Kevin Shinpaugh, Director Research/Cluster Computing,
http://www.computing.vt.edu/research_computing

Hassan Aref, Former Dean and Reynolds Metals Professor, Engineering Science and Mechanics Department
http://www.esm.vt.edu/php/person.php?id=10095

Erv Blythe, Vice President of Information Technology
http://www.it.vt.edu/administration/blythe.html

Jane Talbot, Photo Librarian, Visual Communications, The Visual and Broadcast Communications Department

"I'd also like to thank the many other administrators, contractors and volunteers that made this all possible. We may have been the glue that held it all together, but everyone who participated gave 110% to this project. Without their dedication and attention to detail the project would not have been possible" –Jason Lockhart

MT

About The Author

Emmanuel Stein has been an avid Mac user since 1984 and has honed his cross-platform skills while working at several Fortune 100 companies. He has recently started his own Mac-centric consulting company, MacVerse, which offers implementation, system administration and development services geared towards the enterprise market. You may reach him at estein@macverse.com.

PROTECT YOUR POD

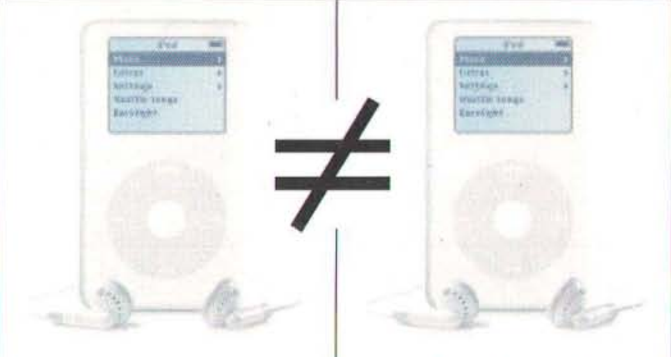
These lightweight super-tough aluminum cases for the iPod® are the most durable and stylish EVER!

Customized engraving and other colors available.

www.A1QP.com

Designed and manufactured in the USA by A-1 Quality Products, Inc.

What's the ***difference***?



Brand new. It's perfect. It's beautiful...

and it's *vulnerable*!

It's naked, exposed to the elements.

It's...

INHUMANE

and it ***will*** get scratched.

Brand new. It's perfect. It's beautiful...

and it's **protected**!

Invisible, durable, & rugged protection.

SAVE the iPod

www.the***invisible*****SHIELD**.com

Trail Vue™ for iPod video

Rugged yet stylish protection

Includes:
- iPod case
- Removable belt clip
- Five interchangeable cords
- Fits all iPod video models

You didn't compromise on your music player—don't compromise on your case.

www.MARWARE.com

NAGIOS ON OS X, PART 2

SETTING UP NAGIOS 2.0

Since part one of this article was published, there have been some changes in Nagios, (the reasons behind the delay for part two in fact.) Nagios 2 has reached the release candidate stage, and as such, I felt that this part should deal with what (should) be the current version by the time you read this. Luckily, this doesn't really change anything in part 1 other than the version of Nagios you download and install, and one minor change for the configure step of installing Nagios.

Addendum and Errata

The change involves an additional group, the nagios command group. I use `nagioscmd` for the name of this group, and you create it as you did the `nagios` group in part 1. This brings us to a rather obvious error in part one, and one I should have caught. The configure command in part one is incorrect, and if it works at all, will give you an incorrect setup. With Nagios 2 in mind, the correct configure command is:

```
./configure —with-gd-lib=/opt/local/lib —with-gd-
inc=/opt/local/include —prefix=/usr/local/nagios —with-
cgiurl=/cgi-bin —with-htmlurl=/ —with-nagios-user=nagios
—with-nagios-group=nagios —with-command-group=nagioscmd
```

That should work correctly for you. The rest of part one should be unchanged for you, it has been for me.

Initial Configuration

One change in Nagios 2, and one that will be welcomed by administrators new to Nagios is the initial configuration. In Nagios 1.2 and later, you had to use multiple configuration files, and getting them set up, and grasping the relationship between them was a little tricky. With Nagios 2, if you're new to Nagios, or you want to play with a smaller setup before you get into mapping the entire Internet, you can now use a much smaller number of config files (around 3 for a minimal configuration). In fact, you have your choice of two templates to work from, one called "minimal.cfg-sample" and one called "bigger.cfg.sample". As the names imply, they are for minimal, and slightly bigger Nagios setups.

In case I haven't done this, let me stress one critical point: ***This article is not, nor should it be taken as a replacement for the Nagios documentation!*** That documentation is ***far*** more complete, and if this article disagrees with the Nagios docs, the Nagios docs should be assumed to be more correct, unless they are assuming !Mac OS X.

The Config Files

As I have alluded to, Nagios makes use of text config files for its setup and to do its work. They can seem daunting at first, but they do follow a logical flow and they should not intimidate you in the least. When you initially install the config files, they all have the name pattern of <something>.cfg-sample. When you have a file set up as you like, remove the –sample from the end of the name, and Nagios will be able to use it. Note that in general, any changes to a config file will probably require a restart of the Nagios process for those changes to be used.

Basic Nagios config theory

Before we get into specifics of the files, we need to look at the relationship of things in Nagios, so that we might have a better mental picture of what's going on. The basic 'unit' in Nagios is the **host**. A host is a computer, a router, switch, etc. It's anything that Nagios directly probes. Each host has a series of characteristics, like IP address, and name that define it to Nagios. Since you can have hundreds, if not thousands of hosts in a network, and applying settings and services to them individually would be rather tedious, you have **hostgroups**. Hostgroups are just what they sound like, a collection of hosts that allow you to more easily work with hosts. So you can apply a probe to a single hostgroup instead of 300 hosts. Easier, no?

Now, since in addition to monitoring, we have to notify people we have **contacts**. Contacts are the human versions of hosts. You can use email, paging, whatever method you can script Nagios to use to notify contacts. As we'll see, you can also set up working hours and non-working hours for notifications too, so non-critical notifications aren't paging people at 2am. Like hostgroups, we have **contactgroups**, since in larger organizations, you may have quite a few people who need to receive notifications from the same host or hostgroup.

The actual probes Nagios uses are **checkcommands**, and that's what determines the information that Nagios checks. However, you don't directly apply the checkcommands, rather you use **services**, which use the checkcommand definition, and other parameters to probe host(groups). This makes it easier to have many checks per host.

Another basic concept is the **dependency**. This can be used to make sure that you're not getting spurious alerts. For example, if you have a monitored switch that has 24 monitored hosts connected to it and the switch goes down, you'd potentially get alerts from all 25 hosts and however many services are being monitored on each host. If the switch is down, you can't talk to the hosts anyway, so those alerts are somewhat useless, as what they're really telling you is that the hosts are unreachable, not that they're actually malfunctioning. So, we use dependencies to say "If Switch A is down/unreachable, don't bother me with alerts from its attached hosts." You can have both service and host dependencies, and both are critical to a happy Nagios installation.

Finally we have extended info, for both hosts and services which can be thought of as metadata. This is useful for things like 3-D icons, connecting Nagios to various graphing utilities, etc.

nagios.cfg

The nagios.cfg file is the heart of Nagios. Without it, nothing works. So, understanding it is critical. Luckily, while it's long, it's really pretty simple, and well-documented. (I really have to put in a commendation to the Nagios team for their documentation. It's an excellent example of how to do useful documentation, and far more projects, commercial and open source both could do well to learn from Nagios' example.)

air power

No surgery required!

Works for all models & speeds

Introducing
The Bird Killer

Signal. Strong signal. That's what you'll get from this inventive new wi-fi extending product from QuickerTek. The Bird Killer attaches to the outside of any iBook or Powerbook — no internal tinkering needed! To learn more about this, and other Mac-friendly inventions, visit ***www.quickertek.com***

Also available...

Antennas
Whips | Plug n' Plays
Base Station

Transceivers
Base Station | Mac Mini
Power Mac

Handles
PowerBook | iBook
TiBook

Wireless Performance Products *for Mac by*
QUICKERTEK
www.quickertek.com **GO!**

The nagios.cfg file is really a listing of other configuration files and settings that apply to Nagios as a whole. For example this line:

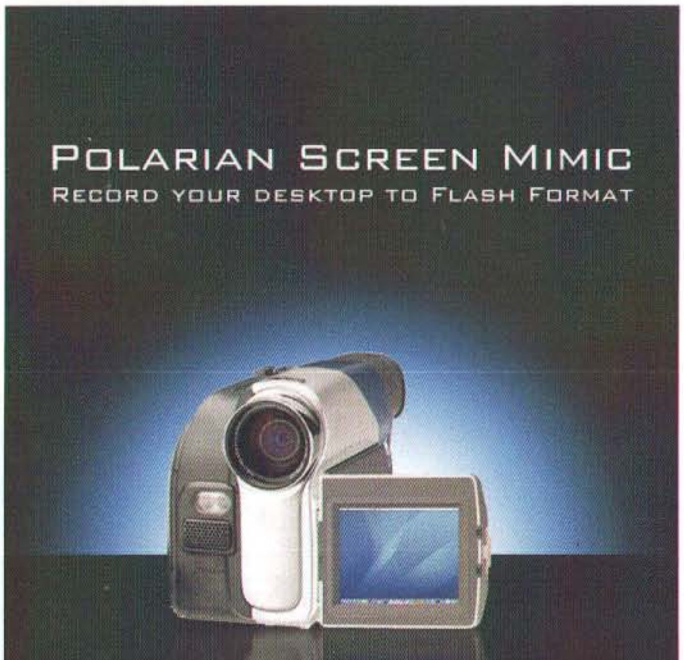
```
log_file=/usr/local/nagios/var/nagios.log
```

Tells Nagios where its log file lives. That's the default from a new Nagios installation. Going down nagios.cfg, we see the entries for the checkcommands file, the misccommands file, and the minimal.cfg file, which is the very basic Nagios configuration file, and handy for new users. As you go down the list, we see that you can get quite complex with the configuration files, giving you the flexibility to grow your Nagios installation to as large as you need to be. (At the high end, you can have clusters of Nagios servers all talking to each other. We're not going to get into those here.)

Other entries in nagios.cfg include the user and group Nagios runs as, and whether external commands are to be used. (note that to use the Web interface, you have to set `check_external_commands=1` in nagios.cfg) One of the nicest things about Nagios is that pretty much every entry in nagios.cfg is nicely commented, so you don't have to guess at what any of them does. Since this is just a basic intro to Nagios, we only need to enable external commands. We'll leave the rest alone.

minimal.cfg

This file is (obviously) a minimal config file that will let you get Nagios up and running with a minimum of work. While it's not going to be one you'll want to use for large installations, it has everything you need to get started.



The first section defines time periods, (can be separately defined in timeperiods.cfg). While minimal.cfg only defines the "24x7" time period, you can create others to suit your own needs, (working_hours, non_working_hours, weekends, etc). The syntax is pretty self explanatory; a 'define timeperiod' block, containing a name used by other config files, an alias that can be more descriptive, contain spaces, etc, and the days in the timeperiod with hours, in 24hr time, that each day covers in the timeperiod, one per line. You can create more if you like, or just use the 24x7 one.

The next section defines the commands Nagios uses to talk to hosts and hostgroups, (can be separately defined in checkcommands.cfg and misccommands.cfg. misccommands is used for things like notification commands and other commands that don't directly use a nagios command plugin). This is where you define the commands that make up services. Looking at this section, we see the checkcommands.cfg file has some commands already set up. We'll skip down past the first two notification commands, to the check-host-alive command, as it's simpler to explain. The basic syntax is simple:

```
# Command to check to see if a host is "alive" (up) 
by pinging it—
a comment for your use
define command{
—start the command definition block
  command_namecheck-host-alive --the name you refer to the command
— as in the rest of Nagios
  command_line  $USER1$/check_ping -H $HOSTADDRESS$ -w 
99.99% -c 100.100% -p 1
—the actual command
  }
```

The command string is pretty straightforward. The `$USER1$` macro, defined in resource.cfg, is the path to the plugin directory, normally /usr/local/nagios/libexec. You can use the $USERx$ macros to define all kinds of commands, like SNMP community strings, etc. You can have up to 32 of them, and they make life a lot easier. The `check_tcp` is the actual command executable name, (in general, Nagios command plugins are of the form ***check_<functionname>***.), followed by various switches. A common one is the "-H" switch, which is the host address. Rather than entering it manually for every host, we use the $HOSTADDRESS$, which is defined in the host entry for minimal.cfg, and we'll see that in a bit. The rest of the switches are command-specific, and are explained by the commands help function. You can bring this up in terminal by running `/usr/local/nagios/libexec/commandname -h` in Terminal, and this will bring up the command's syntax definitions. I've yet to run into a command where this didn't work. Before using a command on a host or hostgroup, it's a good idea to use –h to make sure you know how the command works and what it is going to tell you.

Next up is the contact definition, (also defined by contacts.cfg), which tells Nagios who to notify when it needs to. Since minimal.cfg is by design a simple setup file, there's only one entry here, although you can put more in if you like. The contact_name is how you refer to the contact in the rest of Nagios, the alias is there for more human-friendly labeling. Note that you have separate host and service notification periods. While they're the same in this default, there are cases where you may want them to be different. For example, a backup service that only runs on the weekends wouldn't need 24x7 monitoring, but the host it runs on would. So you could set up a "backup_admin" contact that only received service notifications during a "weekend" time period.

The service and host notification options lines are nothing but a set of switches defined thusly:

- **d** = send notifications on a DOWN state
- **u** = send notifications on an UNREACHABLE state
- **r** = send notifications on recoveries (OK state)
- **f** = send notifications when the host starts and stops flapping
- **n** = no host notifications will be sent out
- **w** = send notifications on a WARNING state
- **u** = send notifications on an UNKNOWN state
- **c** = send notifications on a CRITICAL state

Note that "u" can have different definitions depending what kind of notifications you have. "Flapping" is the Nagios definition for a host that is changing states too often. To avoid this you can enable and configure "flap detection" in nagios.cfg. Flap detection can be quite useful if you have a balky host or service, or if you're having other problems causing hosts or services to look like they're coming up and down a lot.

The service_notification_commands and host_notification_commands lines are how you need to be notified for service and host alerts, (email in this case), and then you have a line for the email address you wish to use for the notifications.

The Contact Groups section follows, (defined separately in contactgroups.cfg), and is, obviously, where you create contact groups. The syntax is similar to the contacts configuration, (you'll note that Nagios uses as many common terms as possible in its config files, which makes things much easier on you) with the "members" line being a comma-delimited list of contacts that will be notified when that particular contact group is notified.

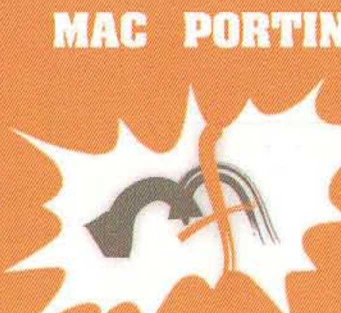
Next up is the Hosts section, (defined separately in hosts.cfg). This is the section where you tell Nagios what to monitor. The host definition has, by necessity, a largish list of terms, even in minimal.cfg:

```
define host{
     use                    generic-host; Name of host
     template to use
     host_name              localhost
     alias                  localhost
     address                127.0.0.1
     hostgroups             test
     check_command          check-host-alive
     max_check_attempts     10
     notification_interval  120
```

ARE YOU MISSING OUT ON SALES

Because your Product / Solution is only on Windows ?

MINDFIRE SOLUTIONS : Your Offshore Partner For MAC PORTING & DEVELOPMENT



- Over 50 Mac projects implemented.
- 5 + Years of Porting experience.
- Tools / Environment : C++, Wx widgets, Visual Studio, Quark QDR, Indesign CS, Acrobat SDK, Mac OS API.
- Platforms : Mac OS 8 & 9, Mac OS X.
- 6 Years old, 200 people strong, ISO 9001:2000 certified.

contact : shankarn@mindfiresolutions.com, 877-204-6575
www.mindfiresolutions.com, www.portinggurus.com

```
notification_period    24x7
notification_options   d,r
contact_groups         admins
   }
```

The use line tells the defintion which template to use, in this case, "generic-host", defined just above the host definitions in minimal.cfg. The templates are handy for defining values that are going to be common to a host or group of hosts, (not a hostgroup), so that your host definitions don't have to be needlessly long.

The host_name is how you refer to the host in the rest of Nagios, the alias is for a more human-friendly label. The address is what is used by the $HOSTADDRESS$ macro we saw in the command definitions earlier. It can be a FQDN DNS name, or an IP address. For servers, I prefer the IP address wherever possible, since that way, the DNS service on my network dropping doesn't kill Nagios' ability to find hosts. One line not in the default, but that I included here is hostgroups. You can, if you like, define a host's hostgroups in the host definition *or* a separate hostgroup definition. I recommend picking one and sticking with it to avoid confusion. The check_command line is a single command that you can use as the basic "is it up or not command". This isn't where you set up all the services you check on a host, we'll look at that later. This is just a default command for a given host. max_check_attempts is how many times Nagios will retry the check_command if the result is anything other than OK. (This only applies to the check command in the host definition, not all services running against that host).

notification_interval is how many time units, (default is minutes) that Nagios will wait to send out notifications of a host that is still down or unreachable. That's continuously down. If the host goes up and comes back down, that's different. notification_period is the timeperiod that notifications for this host are allowed, and use the timeperiod(s) set by you. notification_options determine the conditions that notifications are sent out. Usually, you want at least d,r, so that you know when a host goes down, and if it comes back up by itself. contact_groups are self-evident, they're the groups who get notified. If you want multiple contact groups, then use a comma-delimited list. Please note that this example is not a complete list of host parameters by any means, and you should consult the Nagios documentation for a full list.

Since we just defined the host, we should next define the host group the host belongs to, and that's the next section, (separately in hostgroups.cfg):

```
define hostgroup{
        hostgroup_name    test
        alias             Test Servers
        members           localhost
         }
```

As you can see, hostgroups look a lot like contact groups. The members parameter is a comma-delimited list of hosts if multiple hosts are used.

The final section in minimal.cfg is services, (defined separately in services.cfg). This is where you really get into the meat of Nagios. Services are how you apply commands to multiple hosts with a single entry, and notify multiple contacts or contact groups. Services can be any command Nagios knows about, and you can get quite specific. For example, while there's no specific command to check the KDC status on OS X Server, I was able to do so by using SNMP to check for the KDC process by using the following command definition:

```
#check_kdc_process_via_snmp command definition
define command{
        command_name     check_kdc_process_via_snmp
        command_line     $USER1$/check_proc_by_snmp
$HOSTADDRESS$ $USER3$ $USER9$
        }
```

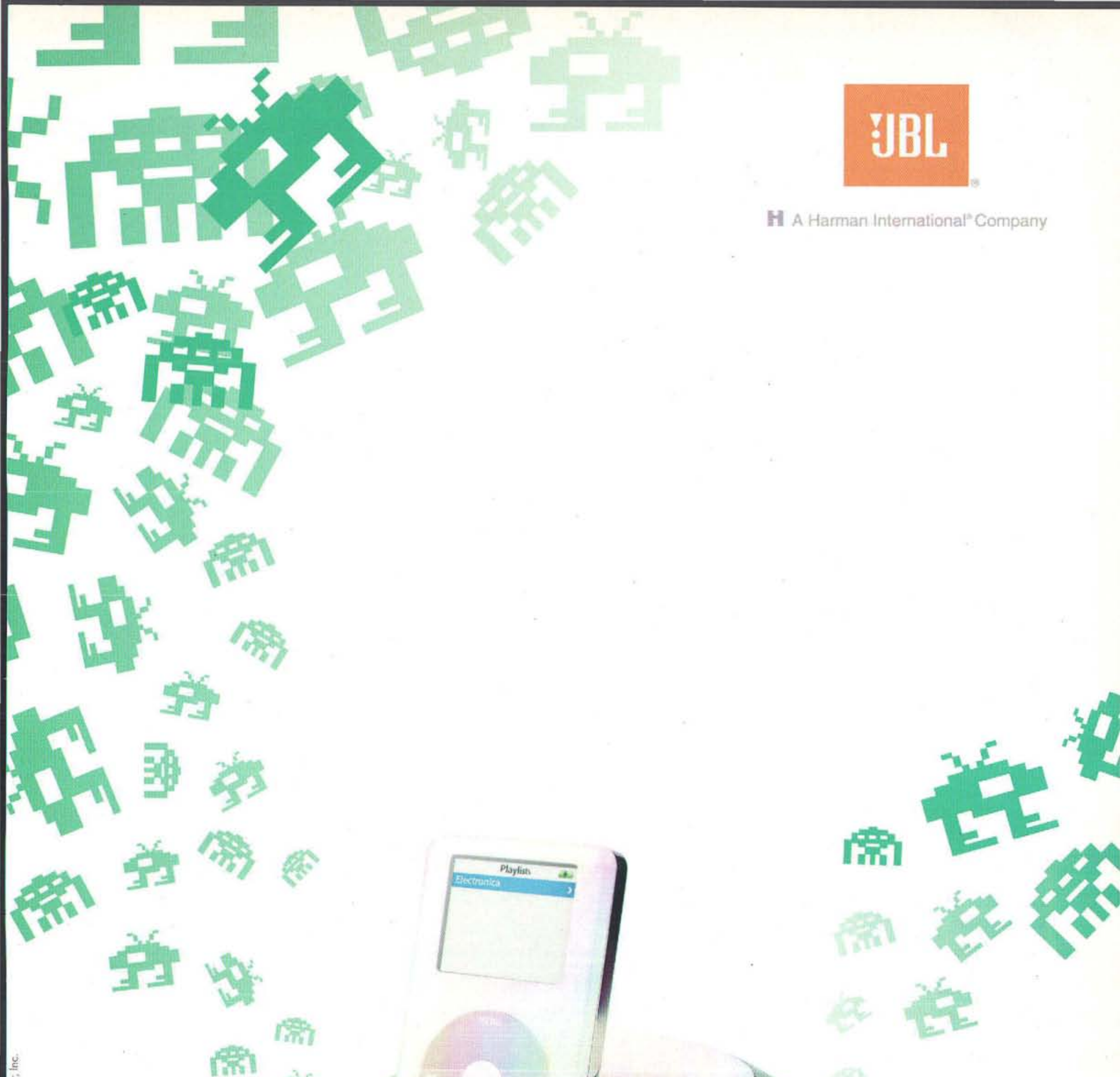
and wrapping it in a service:

```
define service{
        use                              generic-service
; Name of service template to use

        host_name                        xserve01
        service_description              SNMP KDC Process
Check
        is_volatile                      0
        check_period                     24x7
        max_check_attempts               3
        normal_check_interval            3
        retry_check_interval             1
        contact_groups                   xserve-admins,nt-
admins
        notification_interval            120
        notification_period              24x7
        notification_options             w,u,c,r
        check_command
check_kdc_process_via_snmp
        }
```

Like host definitions, service definitions have a template option, so you can set common parameters once and apply them to all the services that use this template. Let's take a look at the default "PING" definition in minimal.cfg:

```
define service{
        use                              generic-service
; Name of service template to use
        host_name                        localhost
        service_description              PING
        is_volatile                      0
        check_period                     24x7
        max_check_attempts               4
        normal_check_interval            5
        retry_check_interval             1
        contact_groups                   admins
        notification_options             w,u,c,r
        notification_interval            960
        notification_period              24x7
        check_command
check_ping!100.0,20%!500.0,60%
        }
```

If we compare the service to the host definitions, we see that they're quite similar. The use parameter is the template the definition uses. The host_name paramter is a comma-delimited list of hosts this service runs giants. The



JBL

A Harman International® Company

iPod® OUTSPOKEN.

JBL ON STAGE™: SHARE MUSIC OUT LOUD WITH THIS HIGH-PERFORMANCE SOUND SYSTEM FOR YOUR MP3 PLAYER.

WWW.JBL.COM/JBLONSTAGE

max_check_attempts, notification_interval, and notification_periods are the same as for the host definition. The is_volatile parameter normally doesn't apply to services, and should be left at its default unless you have a need to change it. The normal_check_interval setting is how many time units to wait between regular checks of a service. By default, this is set to every five minutes. You can increase the frequency of checks, but this will also increase the load on your server and network, and should be done with caution. The retry_check_interval is how long to wait before scheduling a 're-check' which only happens in the case of a non 'OK' return from a check. The contact_groups parameter is exactly the same as for the host definition. The check_command is the name of the command as defined in the checkcommand definition, and any additional parameters the command needs.

That, in a nutshell is minimal.cfg, and almost all the settings you need to get started with Nagios. However, there are still a couple more files to set up before we're ready to start Nagios and get monitoring.

cgi.cfg

This is the config file that controls how Nagios talks to the CGIs and who can access which CGIs. This file isn't too complicated, but it has to be correct for Nagios' web interface to work correctly. Like all Nagios files, it's well commented. Running down the entries, most are self explanatory, so I won't comment on all of them. One of the ones that can catch you off guard is the url_html_path entry. Remember, with Mac OS X Server, that's going to be the root defined for the site in Server Admin, so you want that to point at the path defined by the physical_html_path parameter just above it.

The use_authentication parameter and the access control sections that follow it are critical, and you really, ***really*** want to read the Nagios documentation on how they work. If you are using a lot of external commands that can reboot hosts, etc, (all possible with Nagios), properly configuring your CGI access controls is ***critical*** to the security of your Nagios installation.

The statuswrl_include parameter is if you want to create a 3-D VRML 'flythrough' view of your network. It's not really any more useful than any of the 2-D views, but it's pretty cool for corner office types. The rest of the options can be left alone for an initial installation.

Testing Your Configuration

Of course you read all the docs and did everything right, but just in case, Nagios gives you a way to test your config, via the –v option for the nagios executable. The syntax is <path to the nagios executable> -v <path to nagios.cfg>. So for our example, we'd use:

```
/usr/local/nagios/bin/nagios -v
/usr/local/nagios/etc/nagios.cfg
```

If we did everything right, Nagios will tell us, and we can start it up. If there are config file errors, Nagios will do its best to give you the file name and the line number with the error. This info has always been fairly accurate in my experience, so just look where Nagios tells you, and you should be able to find any errors quickly.

If you didn't get any errors, then let's start nagios as a daemon. This is done by using the –d switch as below:

```
/usr/local/nagios/bin/nagios -d
/usr/local/nagios/etc/nagios.cfg
```

Once that's done, run top to make sure Nagios is running. If it is, congratulations, you have a working Nagios installation. If not, run Nagios with the –v option to see what you may have missed. Checking system.log can help here as well.

Conclusion

Well, we've gone over a ***very*** basic Nagios configuration setup guide, (and corrected some errors from part 1). In the third part of this, we'll take a look at the actual interface, and get an idea of what we're looking at when we check on Nagios, along with some of the email notifications you might get. Thanks!

Bibliography and References

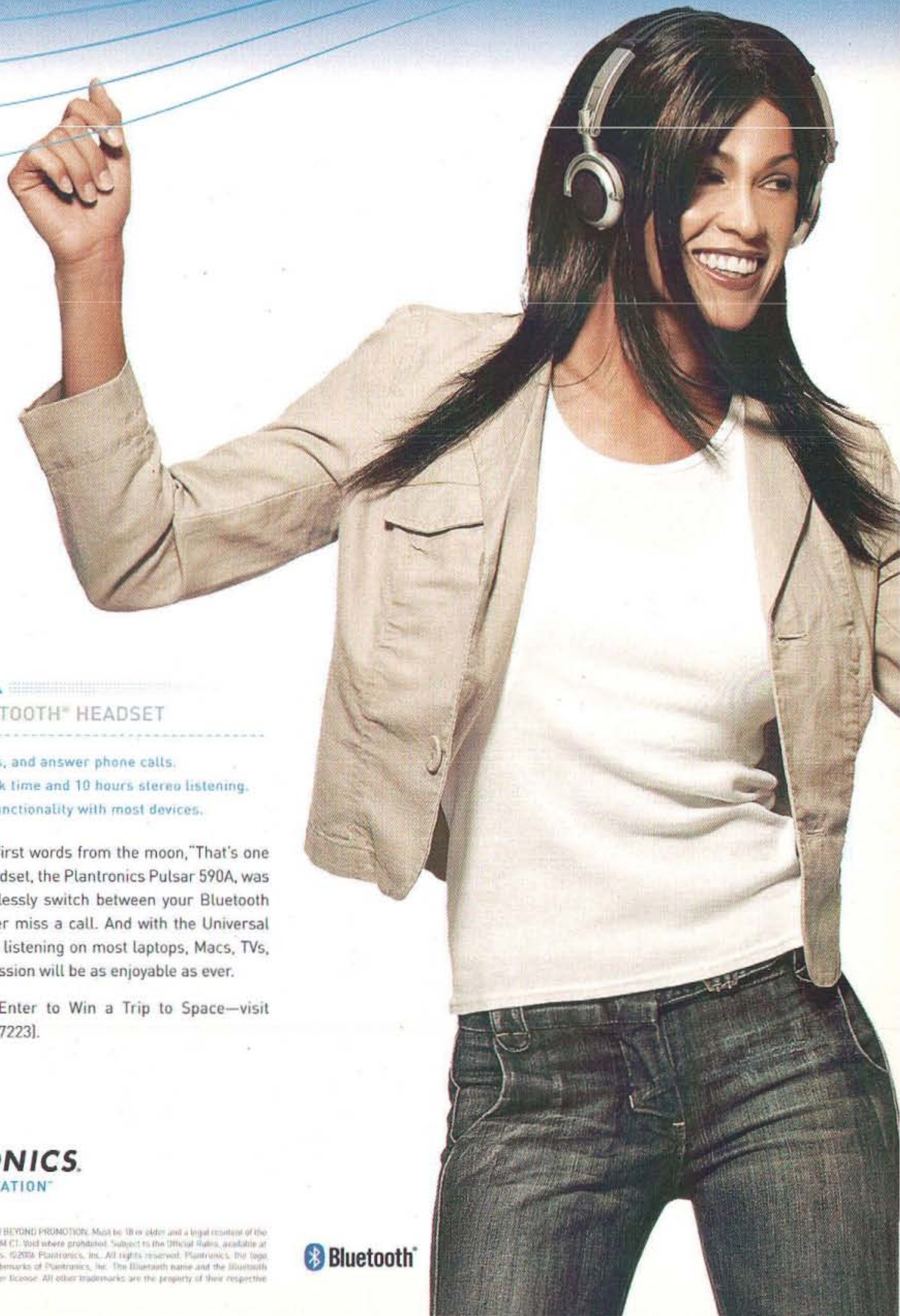
There are two sites that you really must get familiar with to use Nagios. http://www.nagios.org/ is the main Nagios site, and has tons of excellent information for you to use. The other is http://nagiosexchange.org, the biggest collection of Nagios plugins you'll find anywhere.

MT

About The Author

John Welch <jwelch@bynkii.com> is Unix/Open Systems administrator for Kansas City Life Insurance, (http://www.kclife.com/) a Technical Strategist for Provar, (http://www.provar.com/) and the "GeekSpeak" segment producer for Your Mac Life, (http://www.yourmaclife.com/). He has over fifteen years of experience at making Macs work with other computer systems. John specializes in figuring out ways in which to make the Mac do what nobody thinks it can, showing that the Mac is a superior administrative platform, and teaching others how to use it in interesting, if sometimes frightening ways. He also does things that don't involve computertry on occasion, or at least that's the rumor.

Our headsets brought the world closer to the moon.
Now they can keep you close to your music.



PULSAR™ 590A

ULTIMATE STEREO BLUETOOTH® HEADSET

MUSIC AND MOBILE - Listen to music or movies, and answer phone calls.
TALK AND LISTEN LONGER - Up to 12 hours talk time and 10 hours stereo listening.
UNIVERSAL ADAPTABILITY - Enjoy Bluetooth functionality with most devices.

In 1969, a Plantronics headset carried those first words from the moon,"That's one small step for man..." Our latest Bluetooth headset, the Plantronics Pulsar 590A, was designed for versatility here on Earth. Seamlessly switch between your Bluetooth phone and your favorite music, so you'll never miss a call. And with the Universal Adapter, you can experience Bluetooth stereo listening on most laptops, Macs, TVs, DVD players and MP3 players. So your next mission will be as enjoyable as ever.

We've been to space, now it's your turn. Enter to Win a Trip to Space—visit Plantronics.com or text "headset" to SPACE (77223).

PLANTRONICS.
SOUND INNOVATION™

Bluetooth®

Writing a Menulet Extension

Extending an Application or Providing a Service via a Menulet

By Andrew Turner

Introduction

Most modern operating systems have some constantly viewable area that displays useful icons, data, and notifications relating to your computer and other services. Windows uses the 'task tray', the Linux freedesktop.org specifies a 'system tray', and Mac OS X uses the menubar. On the Mac, this menubar is where users see the date/time, audio volume, spotlight icon, and any other number of tools they can install to display information.

The menubar icons and tools are frequently called 'menulets', or status items. Menulets can be an application unto themselves, such as MenuMeters (http://www.ragingmenace.com/ software/menumeters/), which displays the current CPU, memory, or network usage as graphs and blinking lights. A menulet can also extend a larger application such as DesktopManager's (http://desktopmanager.berlios.de/) menulet that allows a user another means to switch between multiple virtual desktops and can display the current viewable desktop.

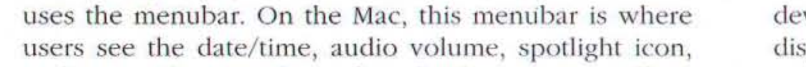


Figure 1: Example Menubar with Menulets.

Figure 1 shows an example menubar with the menulets (from left to right): DesktopManager, Salling Clicker, Applescript Menu, MissingSync, MenuMeters, Date/Time, Audio, and Spotlight.

There are two major 'flavors' of menulets: the standard and publicly documented *NSStatusItem* and the private and undocumented *NSMenuExtra*. *NSStatusItems* are fully capable of doing most anything a menulet can do. The one major benefit of an *NSMenuExtra* is the ability to mouse-drag reorder the menulet with regards to the other system menulets, whereas *NSStatusItems* are placed further along the left-side of the menulets area of the menubar. Because the *NSMenuExtra* API is undocumented, and subject to change by Apple whenever deemed necessary, it is best to avoid using the API in your applications.

The rest of this article will lead you through developing your own *NSStatusItem* menulet including displaying text or an icon in the view, making a drop-menu and setting up automatic updating.

Creating the Menulet

The first thing to do is create a new XCode project for building this tutorial. Alternatively, you could add these files and resources to an already existing application. Menulets must belong to an application since the status bar will not retain and update your menulet for you. Therefore, for purposes of illustration and testing we will be creating a new, stand-alone application for controlling our menulet.

The example menulet will display the user's external IP address in the menu bar. The external IP address is obtained by querying a remote server and then setting the text in the menubar to the external IP address.

In XCode select *New Project...* and choose to create a *Cocoa Application*. After the new project is created, right click on the *Classes* folder and choose *Add > New File...* You will be adding an 'Objective-C class', and name it "IPMenulet.m". Make sure to leave 'Also create IPMenulet.h' checked.

The application will need to hold a reference to the *NSStatusItem*, as well as initialize the menulet, add it to the system status bar, and update data in the menulet. Add the following code to the *IPMenulet.h* and *IPMenulet.m* files you created in your project.

IPMenulet.h

```
@interface IPMenulet : NSObject {
    NSStatusItem *statusItem;
}

-(IBAction)updateIPAddress:(id)sender;
```

IPMenulet.m

```
-(void)dealloc
{
     [statusItem release];
    [super dealloc];
}
- (void)awakeFromNib
{
  statusItem = [[[NSStatusBar systemStatusBar]
    statusItemWithLength:NSVariableStatusItemLength]
    retain];
  [statusItem setHighlightMode:YES];
  [statusItem setTitle:[NSString
      stringWithString:@"0.0.0.0"]];
  [statusItem setEnabled:YES];
  [statusItem setToolTip:@"IPMenulet"];

  [statusItem
setAction:@selector(updateIPAddress:)];
  [statusItem setTarget:self];
}
```

The code above creates and attaches a new *NSStatusItem* of variable length to the system status bar. Since the status item is variable length, the area will grow and shrink depending on the content of the status item's title or icon. Alternatively, you could set the length via the *statusItemWithLength:* method to *NSSquareStatusItemLength*, which would fix the width of the status item to the height of the status bar (currently always 22 pixels high).

The parameter ***highlightMode*** specifies if a box is drawn around the menulet when clicked, and ***enabled*** determines if the menulet is grayed out or darkened. Since we don't yet know our IP address, we will set the initial display to *0.0.0.0*, and the ***tooltip*** is what is displayed when a user hovers their mouse over the menulet.

At the end of the initialization code there are the ***setAction:*** and ***setTarget:*** functions. Similar to other applications, these functions set the functions that should be called when a user clicks on the menulet. In this example, we will be updating the IP address whenever the user clicks on the menubar. It is also possible to set the double-click action if desired.

We now need to implement the ***updateIPAddress:*** method.

IPMenulet.m

```
-(IBAction)updateIPAddress:(id)sender
{
  NSString *ipAddr = [NSString
stringWithContentsOfURL:
      [NSURL URLWithString:

@"http://highearthorbit.com/service/myip.php"]];
  if(ipAddr != NULL)
      [statusItem setTitle:
        [NSString stringWithString:ipAddr]];
}
```

MACTECH® *domains*

Register

Get your .COM or any other domain name here!

FREE with every domain:

- **FREE! Starter Web Page**
- **FREE! Getting Started Guide**
- **FREE! Complete Email**
- **FREE! Change of Registration**
- **FREE! Parked Page w/ Domain**
- **FREE! Domain Name Locking**
- **FREE! Status Alert**
- **FREE! Total DNS Control**

Just visit

www.mactechdomains.com

to register for your domain today!

Starting at $3.99

The *updateIPAddress:* function gets the IP address as a string from an external server query and then sets the title of the *NSStatusItem*.

Building the NIB Interface

We now have all of the code that we need . However, when the application starts, it needs to know to create the menulet. Therefore, we will create an instantiation of the menulet in the NIB file that is loaded. This will allow us to later add more code and hooks to our menulet for controlling menu items.

To create the menulet double-click the *MainMenu.nib* in the *Resources* folder of the project. After it loads, click the "Classes" tab, then choose "Classes" in the menubar, "Read Files…", navigate to *IPMenulet.h* and "Parse".

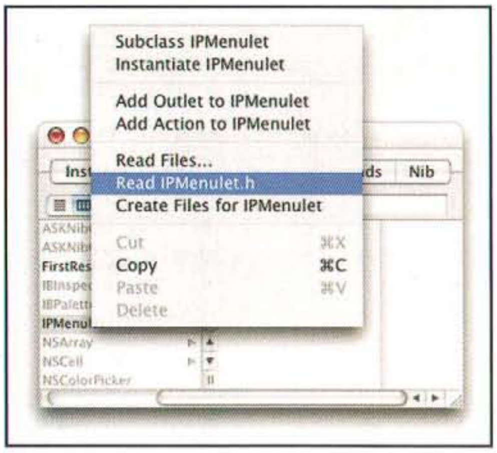


Figure 2: Parsing IPMenulet.h into Interface Builder allows it to be instantiated as part of the application.

This parsing loads the *IPMenulet* interface into Interface Builder. We still need an instance of the class to use in our application. Right-click "IPMenulet" in the pane view and click "Instantiate IPMenulet". This will place a blue-cube and *IPMenulet* instance in our Interface Builder window.

At this point we don't want a main application window to pop-up at startup, so click the *Window* icon and delete it. Then save the nib file, return to XCode and choose the "Build and Go".

Depending on how many menulets you already had in your status bar you may have to click outside of XCode since XCode has a large number of menubar items. You should see *0.0.0.0*. Click on the menulet and after a short pause you should see an updated IP address.

Removing the Dock Icon

If you are making a menulet-only application, having a dock icon seems like an unnecessary item. It is possible to have an application 'hide' its dock icon by specifying a new value in the "Info.plist" file in your XCode project. At the end of the list, just before, the *</dict>*, add the following key and value:

```
<key>LSUIElement</key>
<string>1</string>
```

Now when your menulet application starts up, it doesn't have a dock icon. Be careful however, as you currently have no easy means to quit your menulet application either. While developing, it is possible to quit the application by pressing the "Stop" icon in XCode.

MacResource
the.mac.resource

Wireless Products

Airport Cards
Extreme: $69.99
Standard: $159.99
Routers: $59.99

Accessories

3RD PARTY Wireless Mice $19 to $39
Apple Mice or Kbds $19 to $59
Kensington 4-btn Mice: $19

Parts for Every Mac System!

Logic Boards

G3 B&W Rev2: $99
G4 PCI : $129
G4 AGP Graphics: $219
G4 Gigabit: $349
G4 Digital Audio: $329
G4 Quicksilver: $499
G4 Mirrored Doors: $489
iMac Logics from $199
Beige G3 Series: $49

Power Supplies

Beige G3 Desktop: $49
Beige G3 Tower: $69
Blue & White Tower: $89
G4 Quicksilver: $179
G4 Digital Audio: $189
G4 Gigabit Ethernet: $199
G4 AGP Series 237w: $219
G4 AGP Series 208w: $179
G4 Mirrored Doors: $199

Logic board and power supplies require exchange.

Thousands of parts in stock for all Mac systems!

www.mac-resource.com

Mac Systems

G4/466Mhz: $249
G4/733Mhz: $699
G4/1000Mhz: $879
G4/1.25 OS9:$1099

New Systems Arriving Daily! Call for latest Stock.

eMac 700Mhz: $399
eMac 800Mhz: $449
eMac 1GHz: $629

iMac G4/700MHZ 15": $499
iMac G4/1.0Ghz 17": $749
iMac G4/1.25GHZ 20" $899

Xserves!!!

G4 1.0 ghz DP 512/60gb/CD NO SW	$899
G5 2.0 ghz 512/80gb/CD	$1999
G5 2.3 ghz DP 1024/80gb/Combo	$2999
G5 2.3 ghz DP Nodes from	$1999
Xserve Raid 1.2 TB 7 X 180gig	$3399
Xserve Raid 2.5 TB 14 X 180gig	$4499
Xserve Raid 2.8 TB 7 X 400gig	$4999
Xserve Raid 5.6 TB 14 X 400gig	$6999

Cache, Back-Up Batteries & Drive Upgrades available on all units
Free Ground Shipping when you mention MacTech Magazine

Refurb Displays

Apple 17-inch LCD: $349
Apple 15-inch Studio LCD: $149
Apple 22-inch Cinema: $999
Apple 17-inch Studio CRT (VGA OR ADC): $49

Products are open box, refurbished or demo call for more information

1-888-Mac-Resource

LOOK FAMILIAR?

Get Rid of Your Fax Machine!

Receive your faxes directly to your email account as PDF file attachments

Service available in over 150 area codes

To check out special offers for MacTech readers, visit www.MaxEmail.com/MacTech

maxemail®

Using a Menulet Icon

You now have a fully functional menulet application that takes up a lot of valuable menubar space. For some menulets, it may be best to use an icon that the user will click for more information, or an icon that changes depending on the state of some application variable or other data source.

Unicode Glyphs

The simplest way to add an iconic view to the menulet is by using a Unicode character for the title. This has several benefits, such as not requiring the design, creating, and loading of an image. Also, when a user clicks on a menulet and it becomes highlighted, the icon should have an 'inverse' view that inverts the colors of the icon in some way to alert the user that the menulet has been clicked. The title of a status item, and therefore a symbolic glyph, is automatically inverted.

Finding a good glyph (symbol) is a bit tricky. In the end, we need the hex value for a Unicode character. Mac OS X provides a very nice tool for looking at Unicode characters, though it is a bit buried.

Go to ***System Preferences***, ***International*** and check ***Character Palette***. After you do this you will see your current input language flag in the menubar (look, another menulet!). Click on the flag and choose ***Show Character Palette***. The character palette window pane will pop-up and sit above all of your other windows. At the top of the pane, there is ***View:*** and a drop menu. Click on the drop menu and choose ***Code Tables***. You can now navigate through all of the Unicode characters. When you find a glyph that you want to use, click on it and then get the ***Unicode:*** value next to the zoomed in view of the character.

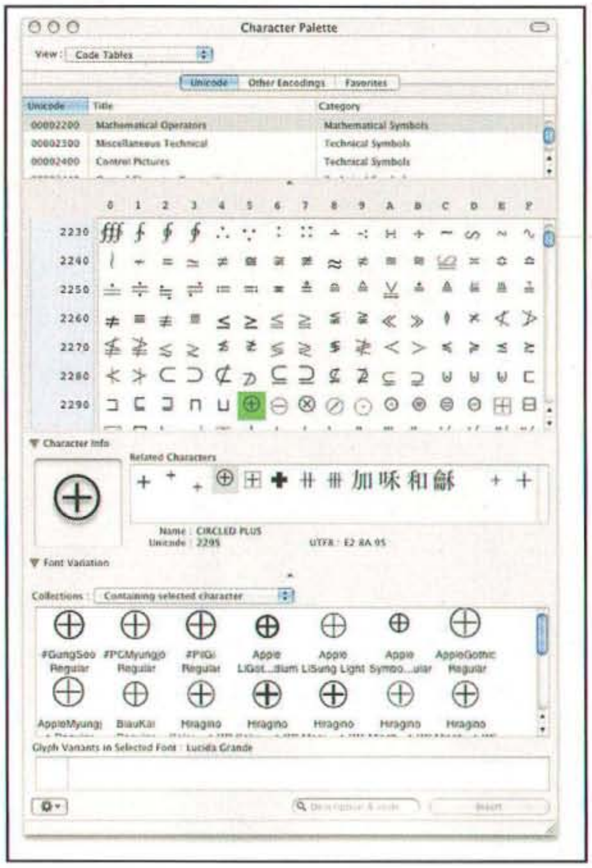

Character Palette
View: Code Tables
Unicode
Other Encodings
Favorites
Unicode
Title
Category
00002200 Mathematical Operators Mathematical Symbols
00002300 Miscellaneous Technical Technical Symbols
00002400 Control Pictures Technical Symbols
Character Info
Related Characters
Name: CIRCLED PLUS
Unicode: 2295
UTF8: E2 8A 95
Font Variation
Collections: Containing selected character
Glyph Variants in Selected Font: Lucida Grande
Insert

Figure 3: The Character Palette is a very useful tool for finding glyphs and their associated Unicode hex values.

MACTECH®
Printer Supplies

Get OEM printer ink, and toner cartridge quality at a fraction of the cost !!!

Inkjet and Laser Toner Cartridges That Create Professional Looking Documents.

We offer high quality 100% compatible inkjet cartridges for Epson, Canon, Brother, and Xerox printers. Black inkjet cartridges from $3.99, color cartridges from $5.99. We also offer Genuine OEM HP, Canon, Lexmark, Sharp, and Xerox printer ink cartridges and laser toner cartridges at discount prices.

http://www.mactech.com/supplies/

YOUR ULTIMATE 3D AQUARIUM!

FORTY BEAUTIFUL FISHES,
TURTLES, JELLIES, & SHARKS!
INTERACTIVE SCREENSAVER
TAP THE GLASS, FEED THE FISH

Allume Systems
A Division of Smith Micro Software

virtual aquarium so vibrant, you'll swear the fish are real! Available from your favorite retailers and catalogs.

Aquazone Seven Seas Deluxe includes 40 fresh and saltwater fish, jellyfish, adult and baby sea turtles, and sharks! Have entire schools of fish on screen swimming through 20 customizable 3D tanks!

Now armed with a Unicode value, return to the *IPMenulet.m* file and replace the

```
[statusItem setTitle:[NSString
  stringWithString:@"0.0.0.0"]];
```

with a formatted string using one (or many) hex values the formatter is *%C*. Therefore, the following code will produce the ??symbol:

```
[statusItem setTitle:[NSString
  stringWithFormat:@"%C",0x2295]];
```

Image Icons

While using a built in glyph is very simple, it does not offer the flexibility an icon provides. The ***NSStatusItem*** API provides the ***setImage:(NSImage*)*** function to set the icon. The following code obtains a 22x22 pixel image that has been added to the project resources, which is then stored and set to the status items image. You can use any small icon in your project that you want. For purposes of this example, the icon is "IPMenuIcon.tif".

First, add a pointer to an ***NSImage*** to the data members of the ***IPMenulet*** class.

IPMenulet.h

```
NSImage *menuIcon;
```

Then we need to get the bundle, the path, and finally the image at this path. We will store this image for future reference if we want to show or hide the icon depending on the menulet's state. We also set the title to an empty string although it is allowed to have both an image and text in the status item. Make sure to release the icon when the application quits.

IPMenulet.m (awakeFromNib:)

```
NSBundle *bundle = [NSBundle bundleForClass:[self
class]];
NSString *path = [bundle pathForResource:@"IPMenuIcon"
ofType:@"tif"];
menuIcon= [[NSImage alloc] initWithContentsOfFile:path];

[statusItem setTitle:[NSString stringWithString:@""]];

[statusItem setImage:menuIcon];

(dealloc:)
[menuIcon release];
```

Test this new version. You should see the icon in the header. When the icon is clicked, the IP address is filled in to the right of the icon as shown in Figure 4.

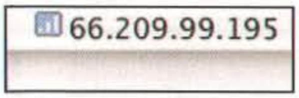

66.209.99.195

Figure 4: The Menulet with both an icon and text title.

Adding a Menu

The menulet has shaped up rather nicely, but is limited to a single line of information and a single action when clicking. It is possible to extend the menulet by adding a menu to provide more information and interaction with user such as bringing up "About" boxes, "Preferences...", or displaying the IP address in the menu and keeping the menulet itself to a single icon.

First we will modify our code to include the pointer to the menu and a menu item we will be creating by adding the following to the ***IPMenulet*** interface definition.

IPMenulet.h

```
IBOutlet NSMenu *theMenu;
NSMenuItem *ipMenuItem;
```

Now we need to build the actual menu. Double-click the ***MainMenu.nib*** file to bring up Interface Builder. The first thing to do is to update Interface Builder's knowledge of the ***IPMenulet*** class. We need to double click the IPMenulet instantiation, right-click and choose "Read IPMenulet.h". After this is complete, return to the "Instances" tab.

Choose the "Cocoa-Menus" tab from the Interface Builder palette. Drag and drop the menu icon in the lower-right corner to the ***MainMenu.nib*** panel. This will add an ***NSMenu1*** item to our nib file. Rename the first item "External IP" and delete the second menu item.

Next we connect the menu to our class. Control left-click and drag from the ***IPMenulet*** instantiation to the new ***NSMenu1*** and connect ***theMenu*** data member of the class.

We assigned the menu to the class, but now the menu now needs to be assigned to ***IPMenulet***. Return to XCode and modify the ***IPMenulet.m*** file to use ***theMenu*** instead of calling the ***updateIPAddress:*** selector. We're also going to dynamically create a menu item that is the external IP

Rumpus FTP

The Complete Internet File Transfer Server For Macintosh

Turn any Mac into a file transfer server...

Simple installation & setup

Dozens of advanced features

Fast, efficient server

Local or remote administration

To let anyone send or receive files...

Transfer files with any Web browser

Compatible with all major FTP clients

Support clients on Mac or PC

Download A Free, Fully-Functional Trial Copy Today

http://www.maxum.com/

PHONEPIPE™ VOIP Service

Residential Plans

The PhonePipe 500

$14.99

500 Minutes - US and Canada

*Additional Minutes 3.5 cents

The PhonePipe 900

$19.99

900 Minutes-US, Canada, Austria, Belgium, Chile, China, Denmark, France, Germany, Hong Kong, Ireland, Italy, Malaysia, Netherlands, New Zealand, Norway, Singapore, South Korea, Spain, Sweden, Switzerland, Taiwan, UK

*Additional Minutes 3.5 cents

The PhonePipe Unlimited

$24.99

Unlimited Minutes US and Canada

The PhonePipe Unlimited International

$34.99

Unlimited Minutes-US, Canada, Austria, Belgium, Chile, China, Denmark, France, Germany, Hong Kong, Ireland, Italy, Malaysia, Netherlands, New Zealand, Norway, Singapore, South Korea, Spain, Sweden, Switzerland, Taiwan, UK

*All Plans Include Caller ID, Voicemail, and Three-Way Calling

Business Plans

Base Line

Starting at
$20.00 a Month

Includes:
- VM • CallerID
- 3-Way Calling
- Call Forwarding
- Call Waiting Plus

*Long Distance Starting at 2 Cents Per Minute

Enhanced Line

Starting at
$30.00 a Line

Includes:
- Basic Line Features
- Microsoft Outlook Integration
- Find Me-Follow Me
- Simultaneous Ring
- Personal Web Portal

*Long Distance Starting at 2 Cents Per Minute

Unlimited Line

$49.99 a Line

Includes:
- Enhanced Line Features
- Unlimited Long Distance to US and Canada

*Long Distance Starting at 2 Cents Per Minute

*All Business Lines Include PBX Features: Call Transfer, Music on Hold (Customizable), Call Hold, 4 or 5 Digit Dialing

Optional Features: Auto Attendant, Call Center, Reception Console

www.PhonePipe.com
1-877-300-3035 Ext 8200

address and set it to update the IP address when the user clicks it.

IPMenulet.m (awakeFromNib:)

```
[statusItem setMenu:theMenu];
ipMenuItem = [[NSMenuItem alloc] initWithTitle:@"0.0.0.0"
    action:@selector(updateIPAddress:)
keyEquivalent:@""];
[ipMenuItem setTarget:self];
[theMenu insertItem:ipMenuItem atIndex:1];
```

The menulet will know to call and expand the menu on a user click. Therefore, comment out:

```
// [statusItem setAction:@selector(updateIPAddress:)];
// [statusItem setTarget:self];
```

We also want to update the IP address menu item with the new IP address. This is a simple change, instead of setting the title of the ***statusItem***, we want to change the title of the ***ipMenuItem***.

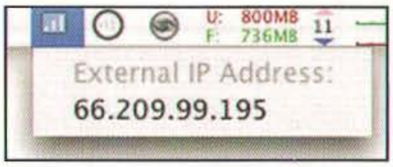
IPMenulet.m (updateIPAddres:)

```
[statusItem setTitle:[NSString stringWithString:ipAddr]];
```

becomes:

```
[ipMenuItem setTitle:[NSString stringWithString:ipAddr]];
```

Click "Build and Go" again to see the newest results. You will only see the icon in the menubar. Click on this to expand down your new menu. Click on the IP Address to have it update and change the menu item.



External IP Address:
66.209.99.195

Figure 5: The menulet with a drop-down menu.

Automatic Updating

To this point we have made a menulet that has either an icon or text in the status bar, and updates the IP address when the user interacts with the menu item. By contrast, most menulets provide the user information without requiring interaction from the user. Therefore, it is often useful to use a timer to automatically update the menulet. This allows the menulet to provide up to date information without requiring constant interaction from a user.

This example hardcodes the timer interval to a predetermined value that seems adequate for the task, 1000 seconds. However, it would be useful for more advanced menulets to include a preference pane or selectable menu items for setting a variable update rate.

Add the following sections of code to your source files:

IPMenulet.h

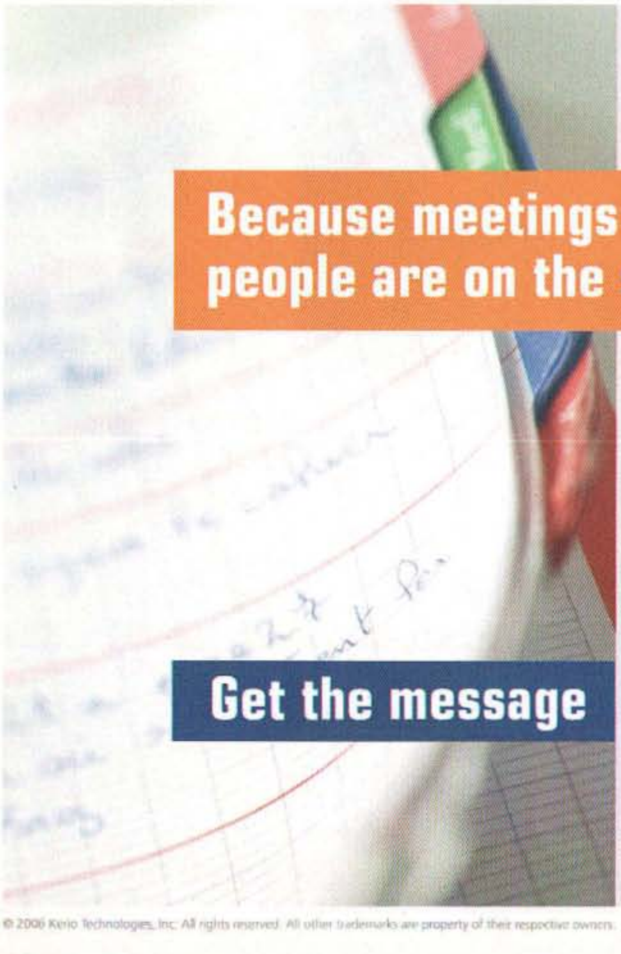
```
NSTimer *updateTimer;
```

IPMenulet.m (awakeFromNib:)

```
updateTimer = [[NSTimer
```

Connect • Communicate • Collaborate • Securely

Because meetings are more productive when people are on the same page... of their calendars



Kerio MailServer

A groupware alternative to Exchange that syncs calendars, contacts and email with Entourage and Outlook. Integrated anti-spam and McAfee virus filtering provide secure, junk-free email for users on any platform.

Mac OS X · Linux · Windows

www.kerio.com

McAfee Proven Security

KERIO

```
        scheduledTimerWithTimeInterval:(1000.0)
        target:self
        selector:@selector(updateIPAddress:)
        userInfo:nil
        repeats:YES] retain];
[updateTimer fire];

(dealloc:)
[updateTimer release];
```

The timer is built with 1000 second delay, and when it 'fires' (timer runs out), it will call our ***updateIPAddress:*** selector. The timer will continue firing for as long as the application remains active (***repeats*** is set to ***YES***).

Build and run the updated application. The IP address will be updated immediately upon startup. It will then be updated every 1000 seconds. If you want to double-check your timer, change the timer interval to 10.0, and add

```
[statusItem setTitle:[NSString
    stringWithString:@"Updating"]];
```

to the beginning of the ***updateIPAddress:*** function. At the end of the function, add:

```
[statusItem setTitle:[NSString
    stringWithString:@""]];
```

These additions will cause the menu to print "Updating" in the menu bar while retrieving the external IP address.

Conclusions

You've now successfully developed several iterations of a Mac OS X menulet. Each variation has different uses, depending on the desired information to be displayed, and user interaction required. Furthermore, menulets as described here can be made either stand-alone or as part of a larger application. In a larger application, the menulet's selectors and menu items would be connected to the application's code and data. This type of interface provides users with a potentially useful resource for quickly gleaning information from your application.

Menulets also provide an interface for drawing to an *NSView*. This capability allows a developer limitless freedom in drawing icons, graphs, blinken-lights, or anything else in their menulet.

For more information on the NSStatusItem API, check out the Apple developer documentation:

http://developer.apple.com/documentation/Cocoa/Reference/ApplicationKit/ObjC_classic/Classes/NSStatusItem.html

MT

About The Author

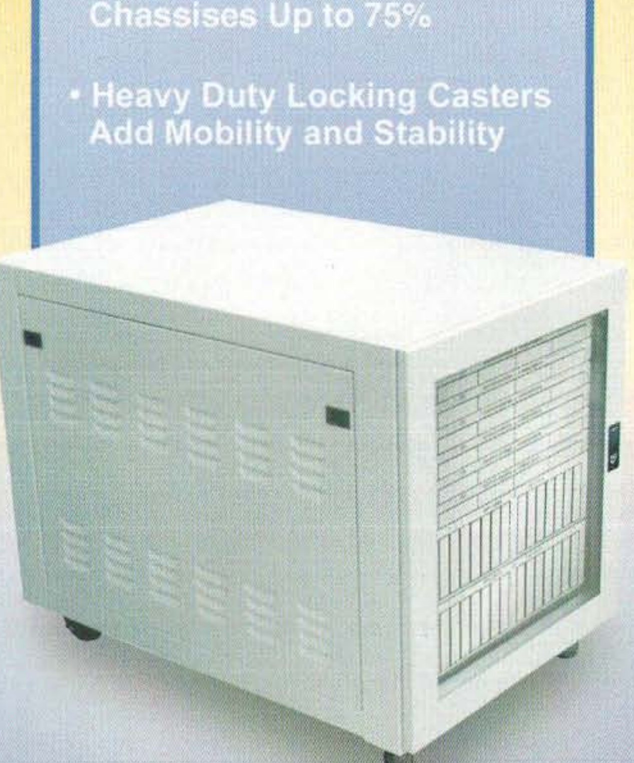
Andrew Turner is a Systems Development Engineer with Realtime Technologies, Inc (www.simcreator.com) and has built robotic airships, automated his house, designed spacecraft, and in general looks for any excuse to hack together cool technology. You can read more about his projects at www.highearthorbit.com.

Keep Your Rack Mount Equipment *Quiet* And Locate It Where You *Need It*

XRackPro2

Noise Reducing Rack Mount Enclosure Cabinet

- Apple Approved and Recommended
- Reduces Noise from Multiple Chassises Up to 75%
- Heavy Duty Locking Casters Add Mobility and Stability



GizMac Accessories, LLC.

XRackPro.com

(800) 475-1677

INTRODUCTION TO DATABASE EVENTS

Data storage and access is an important part of AppleScripting, particularly in complex AppleScript-based projects. Some scripts may need to store user-entered data for later reference, perhaps during an entirely new session. Some may need a location to log activity or errors during processing. Others may need to access structured data, in order to do something fairly complex, such as building a catalog.

In this month's column, we will discuss the use of Database Events, a new and exciting feature in Mac OS X, for storage and access of data during script execution.

Data Storage and Access Option

There are many techniques that are commonly used to store and access data with AppleScript. One common method is to simply write information to a text file on the user's hard drive, on a server, or elsewhere. This can be done with the ***File Read/Write Commands*** suite in the Standard Additions scripting addition, which is installed as part of Mac OS X. This technique of writing information to a text file may suffice for basic types of informational storage. However, depending on the complexity and amount of data being stored or accessed, you may need to write parsing code in order to work with the information.

In AppleScript Studio projects, another method of data storage and access is to make use of a project's preferences file, i.e. it's ***plist*** file. This is actually quite easily done, and is a common technique for storing such information as persistent user interface options and settings.

For complex data storage and access, a more robust solution may be necessary. Typically, developers will turn to a database of some kind. FileMaker Pro is usually quite popular among AppleScript developers, primarily due to its ease of use, customizable interface, and robust AppleScript support. For those not familiar with FileMaker Pro, I highly recommend downloading the 30 day demo from FileMaker's website at http://www.filemaker.com, and checking it out for yourself.

With Mac OS X Tiger, there is a new option – Database Events! If you have never heard of Database Events, that is not surprising. Database Events was released as part of Mac OS X 10.4 Tiger. However, it may have been missed by many, as the only mention of it that I could easily find was a small blurb on the AppleScript webpage at http://www.apple.com/macosx/features/applescript/. In this month's column, we will explore Database Events' syntax, and discuss ways that it can be integrated into your own scripts in order to store and access data during script execution.

What is Database Events?

Database Events is an AppleScriptable background application, which comes installed with Mac OS X 10.4

and higher. It is located in the ***System* > *Library* > *CoreServices*** folder, which you may recognize as the location of other scriptable background applications that are installed with Mac OS X, including Image Events and System Events.

Like its name implies, Database Events provides a way for AppleScript to perform basic events with databases, in particular, SQLite databases. Tasks that can be performed with Database Events include creating, opening, and saving databases, as well as accessing records and fields within databases.

For information about SQLite, visit http://www.sqlite.org/.

By default, Database Events does not appear in the Script Editor's ***Library*** palette, which is accessible via the ***Window* > *Library*** menu. Click the + button in the ***Library*** palette, and navigate to Database Events in order to add it to the palette, thus making it quickly and easily accessible whenever Script Editor is launched. See figure 1.

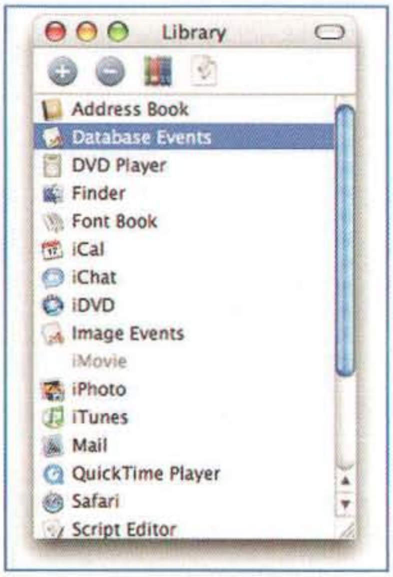
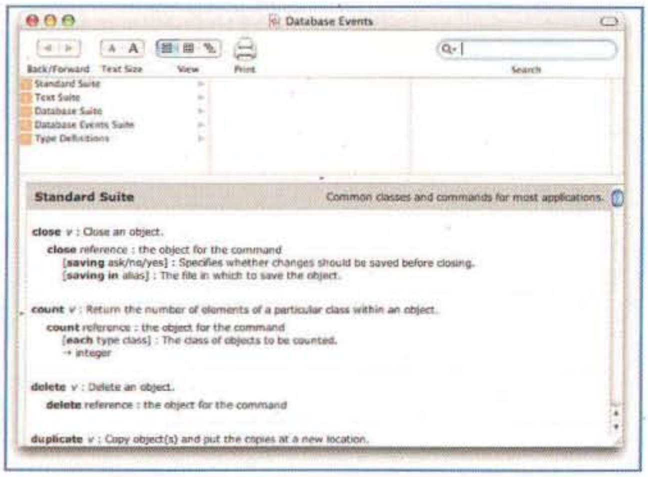


Figure 1. Script Editor Library Palette

Once added to the ***Library*** palette, you may double click on Database Events in order to open its AppleScript dictionary. See figure 2.



Standard Suite Common classes and commands for most applications.

close v : Close an object.
close reference : the object for the command
[**saving** ask/no/yes] : Specifies whether changes should be saved before closing.
[**saving in** alias] : The file in which to save the object.

count v : Return the number of elements of a particular class within an object.
count reference : the object for the command
[**each** type class] : The class of objects to be counted.
→ integer

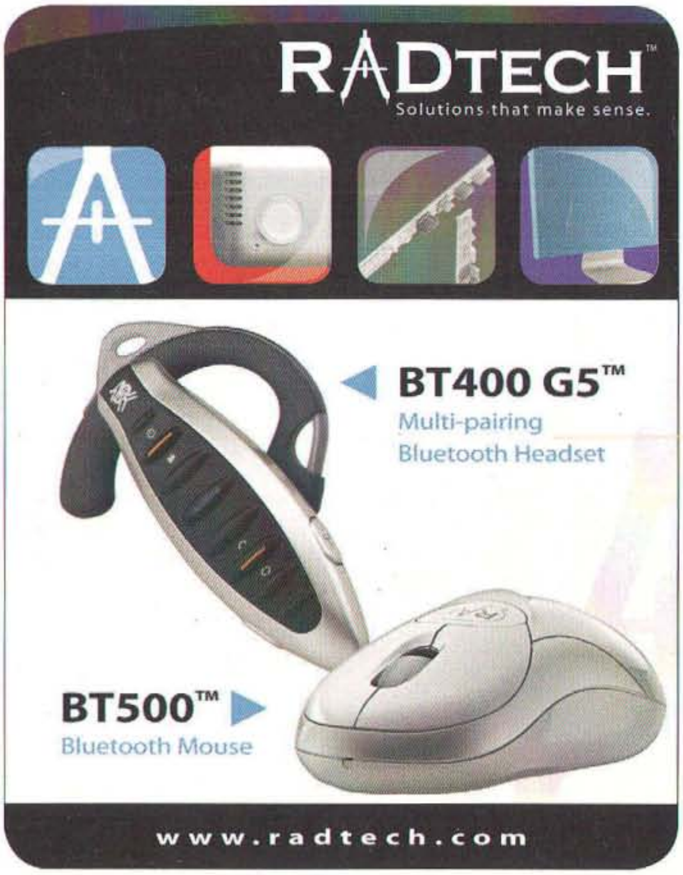
delete v : Delete an object.
delete reference : the object for the command

duplicate v : Copy object(s) and put the copies at a new location.

Figure 2. Database Events AppleScript Dictionary

Getting Started with Scripting Database Events

If you launch the Activity Monitor application, located in the ***Applications* > *Utilities*** folder on your machine, immediately after logging in, you will notice that Database Events is not launched by default. To launch Database Events, use the `launch` command. For example:



```
tell application "Database Events"
  launch
end tell
```

As with any other application, when you are ready to quit Database Events, you may use the quit command. For example:

```
tell application "Database Events"
  quit
end tell
```

Please note that, in order to preserve memory, once launched, Database Events will automatically quit after 300 seconds, i.e. 5 minutes, of idle time or inactivity. When this occurs, it is important to note that any unsaved changes in any opened databases will be lost! Because of this, it is essential that you write your AppleScript code to save any database you may be modifying on a regular basis. We will discuss saving databases a little bit later.

If you require more than 5 minutes of inactivity, then it is actually possible to adjust the delay period by modifying the value of the `quit delay` property of the Database Events application. For example:

```
tell application "Database Events"
  set quit delay to 600
end tell
```

In the example code above, the quit delay is changed from the default 300 seconds to 600 seconds, or 10 minutes. Alternatively, you may also choose to decrease the default delay period by setting the `quit delay` property to a value of less than 300 seconds. Please note that, when modifying the value of this property, it will be reset to 300 seconds again the next time Database Events is launched.

You may check the value of the delay period by retrieving the `quit delay` property. For example:

```
tell application "Database Events"
  quit delay
end tell
--> 300
```

In some cases, you may not want Database Events to quit automatically at all. To disable automatic quitting entirely, set the `quit delay` property to a value of `0`. For example:

```
tell application "Database Events"
  set quit delay to 0
end tell
```

Working with Databases

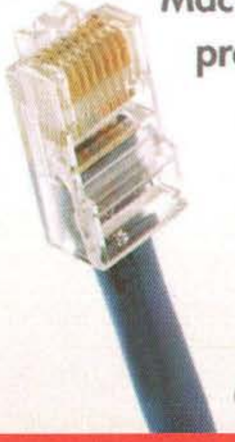
Now that we have discussed the basics of the Database Events application, let's move on to databases.

According to Database Events' AppleScript dictionary, the `database` class is an element of the `application` class. The dictionary also indicates that a database possesses a `name` property and a `location` property.

You can instruct the Database Events application to create a new database by using the `make` command, and specifying a name for the database. For example, the following sample code will create a new database named "Super Heroes", and you can see that the result of the command is a reference to the newly created database.

MACTECH® *Cable Supplies*

Looking for a great selection of cables?



MacTech has selected a quality vendor, with quality products for your cable needs. Benefits include:

- **Lifetime warranty on cables**
- **Same day shipping if you order by 4 PM ET**
- **Express shipping orders can be ordered up to 6:30 PM ET for same day shipping**
- **Huge selection - 3500+ items and growing**

Check out the full selection at the MacTech Cable Store

http://www.mactech.com/cables

E=MC²

Smart

NeoSMART

maxell
Expanding Memory & Mobility

Introducing NeoSMART (Super Maximum-capacity Advanced Reliable Tape) - Maxell's latest technological innovation.

With seven innovative advances, including an ultra-thin tape coating, nanometric metal particles, and superior servo-track writing, NeoSMART is expanding capacities from gigabytes to terabytes and beyond. Providing superior capacity, durability and reliability, NeoSMART is the ultimate solution for the ever-increasing demands of data storage. Just look for the NeoSMART symbol. www.maxell-data.com.

Recordable Media Data Storage Portable Energy Technological Partnerships

```
tell application "Database Events"
  make new database with properties {name:"Super
Heroes"}
end tell
--> database "Super Heroes" of application "Database
Events"
```

Once a database has been created, you can access it by its name, or by its index. For example:

```
tell database "Super Heroes"
```

-- OR

```
tell database 1
```

The following example code demonstrates how to retrieve the properties of an opened database:

```
tell application "Database Events"
  properties of database 1
end tell
--> {class:database, name:"Super Heroes",
location:"~/Documents/Databases"}
```

When accessing the properties of a newly created database, you may notice that the `location` property of the database is set to a value of "`~/Documents/Databases`" by default. This does not mean that the database has actually been created in this location. In fact, this folder may not even exist yet. It simply means that this is the location in which the database will be created, once a `save` command has been issued, or once a record has been created.

Optionally, when a database is created, you may choose to indicate a custom location for the database by specifying a value for the `location` property. For example, the following code will prompt the user to select an output folder. It will then create a new database with a `location` property value of the folder specified by the user.

```
set theOutputFolder to choose folder with prompt
"Please select an output folder:"
tell application "Database Events"
  make new database with properties {name:"Super
Heroes", location:theOutputFolder}
end tell
--> database "Super Heroes" of application "Database
Events"
```

To save an opened database at any time, use the `save` command. This will cause the database to be saved into the value specified in the database's `location` property, using the name specified in the `name` property. For example:

```
tell application "Database Events"
  save database 1
end tell
--> database "Super Heroes" of application "Database
Events"
```

To open a saved database, use the `open` command, and specify the path of the database file to be opened. However, please note that you must specify this path as a POSIX style path. Otherwise, you will receive an error message. For example, the following code will open a database named ***Super Heroes.dbev*** on my desktop.

```
set theDBPath to POSIX path of "Macintosh
HD:Users:bwaldie:Desktop:Super Heroes.dbev"
tell application "Database Events"
  open database theDBPath
end tell
--> database "Super Heroes" of application "Database
Events"
```

To determine the number of opened databases, use the `count` command. For example:

```
tell application "Database Events"
  count databases
end tell
--> 1
```

To close a database, use the `close` command. In order to eliminate the possibility of receiving an error when attempting to close an unsaved modified database, specify a value for the `saving` parameter. For example, the following code will close a database without saving.

```
tell application "Database Events"
  close database 1 saving no
end tell
```

Working with Records

Once a database exists, you can now begin creating records within that database. Like creating databases, the

BLOB XML PHP C++
REALbasic Director COM .NET
SQL Object-Relational Client Server Unicode
Joining Worlds of Information

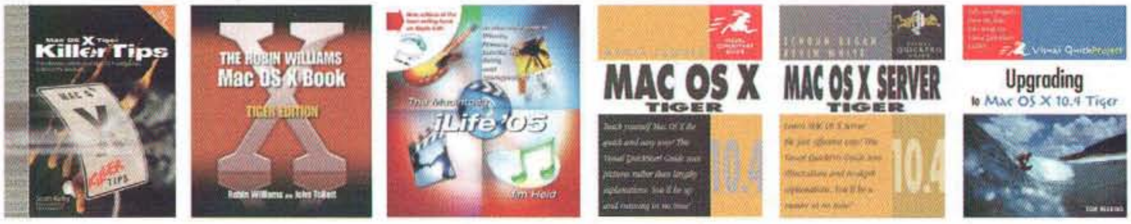
Valentina 2: Deploy True, Royalty Free Client-Server Solutions
www.paradigmasoft.com

PARADIGMA software

Q: What do all these Macintosh all-stars have in common?

Scott Kelby | Robin Williams | Jim Heid | Maria Langer | Schoun Regan | Tom Negrino

A: Peachpit!



Save 35% off the retail price, plus enjoy free domestic U.S. shipping. Visit **www.peachpit.com** and enter in coupon code **EE-FLYR-6787** when you reach the checkout page.
Offer expires 12/31/2006.

www.**peachpit**.com

make command is used to create new records. Records possess an ID property and a name property. A value for the ID property is automatically assigned a unique ID number when a record is created. However, a value for the name property must be specified whenever a record is created. The following example code demonstrates how a new record is created, using a specified name.

```
tell application "Database Events"
  tell database "Super Heroes"
    make new record with properties {name:"Batman"}
  end tell
end tell
--> record id 157660455 of database "Super Heroes"
of application "Database Events"
```

You can see that, in this case, the result of the make command is a reference to the newly created record, and that a unique ID of 157660455 has been applied.

Once a record exists, it may be referred to by its index, its unique ID, or its name. For example:

```
tell record 1

-- OR

tell record id 157660455

-- OR

tell record "Batman"
```

To count the records of a database, use the count command. For example:

```
tell application "Database Events"
  tell database "Super Heroes"
    count records
  end tell
end tell
--> 1
```

You can delete records in a database by using the delete command. For example, the following code demonstrates how to delete every record in a specified database.

```
tell application "Database Events"
  tell database "Super Heroes"
    delete every record
  end tell
end tell
```

Working with Fields

Once a record has been created, you must create fields to be populated within that record. This must be done every time a new record is created. Fields may not be created at the database level in Database Events.

By default, a ***name*** field already exists when a record is created, and it contains the name of the record. You may create additional fields by using the make command. Fields have a name property and a value property, which may be specified at the time of field creation.

The following code demonstrates how to create a record with two fields. The first field, called ***name,*** is created automatically and assigned a value of ***Batman*** when the record is created. The second field, called ***Secret Identity***, is assigned a value of ***Bruce Wayne*** when the field is created.

```
tell application "Database Events"
  tell database "Super Heroes"
    set theRecord to make new record with
properties {name:"Batman"}
    tell theRecord
      make new field with properties {name:"Secret
Identity", value:"Bruce Wayne"}
    end tell
  end tell
end tell
--> field "Secret Identity" of record id 157660455
of database "Super Heroes" of application "Database
Events"
```

To count the fields of a specified record, use the count command. For example:

```
tell application "Database Events"
  tell database "Super Heroes"
    tell record "Batman"
      count fields
    end tell
  end tell
end tell
--> 2
```

Pulling it Together

Now that we have discussed creating databases, records, and fields, let's pull it all together. The following example code demonstrates how to create a new database of records, based on a list of data. The database's name will be ***Super Heroes***, and it will consist of a record for each super hero. Each record will contain a name field, and a secret identity field. Once built, the database will be saved into the default location, the ***Documents > Databases*** folder within your user folder.

```
set theSuperHeroes to {{"Batman", "Bruce Wayne"},
{"Spiderman", "Peter Parker"}, {"Superman", "Clark
Kent"}}
tell application "Database Events"
  set theDatabase to make new database with
properties {name:"Super Heroes"}
  tell theDatabase
    repeat with a from 1 to length of
theSuperHeroes
      set theCurrentSuperHeroInfo to item a of
theSuperHeroes
      set theRecord to make new record with
properties {name:item 1 of theCurrentSuperHeroInfo}
      tell theRecord
        make new field with properties
{name:"Secret Identity", value:item 2 of
theCurrentSuperHeroInfo}
      end tell
    end repeat
  end tell
end tell
```

Now that you have a database, complete with fields and records, you can search for records in a variety of ways. For example, the following code demonstrates how to locate a record by name:

```
tell application "Database Events"
  tell database "Super Heroes"
    first record whose name = "Superman"
  end tell
end tell
--> record id 157661083 of database "Super Heroes" of
application "Database Events"
```

The following code demonstrates how to locate a record by the value of one of its fields:

```
tell application "Database Events"
  tell database "Super Heroes"
    first record whose value of field "Secret Identity"
contains "Parker"
  end tell
end tell
--> record id 157672965 of database "Super Heroes" of
application "Database Events"
```

You can also retrieve and modify the values of fields in existing records. The following example code demonstrates how to retrieve the value of a specified field:

```
tell application "Database Events"
  tell database "Super Heroes"
    set theRecord to first record whose name =
"Superman"
    tell theRecord
      value of field "Secret Identity"
    end tell
  end tell
end tell
--> "Clark Kent"
```

The following code demonstrates how to modify the value of a specified field:

```
tell application "Database Events"
  tell database "Super Heroes"
    set theRecord to first record whose name =
"Superman"
    tell theRecord
      set value of field "Secret Identity" to "Kent,
Clark"
    end tell
  end tell
end tell
```

In Closing

As you can see, there is a lot that can be done with Database Events. I encourage you to begin using it, testing it on your own, and integrating it into your scripts. While Database Events is not as robust as a fully featured database application, it can provide a quick and easy way to store and access structured data.

Also, please be aware that Database Events is currently at version 1.0. As it continues to mature, I am sure that it will continue to grow and will become an important part of many AppleScript-based workflows.

Until next time, keep scripting!

MT

About The Author

Ben Waldie is the author of the best selling books "AppleScripting the Finder" and the "Mac OS X Technology Guide to Automator", available from http://www.spiderworks.com. Ben is also president of Automated Workflows, LLC, a company specializing in AppleScript and workflow automation consulting. For years, Ben has developed professional AppleScript-based solutions for businesses including Adobe, Apple, NASA, PC World, and TV Guide. For more information about Ben, please visit http://www.automatedworkflows.com, or email Ben at applescriptguru@mac.com.

Some people say the G5 iMac is a desktop computer

Our customers think of it as a notebook on steroids.

If you need uncompromising performance 'To Go', call us.

www.ilugger.com • 877-897-5158

LEARNING UNIX FOR MAC OS X TIGER

By Mary Norbury-Glaser

Learning Unix for Mac OS X Tiger is O'Reilly's fourth revision of this popular volume on Unix basics for Mac OS X users. This edition's author, Dave Taylor, is a well-known and highly respected Unix geek with many diverse publications under his belt: **Learning Unix for Mac OS X Panther**, **Creating Cool HTML 4.0 Web Pages**, **Solaris 9 for Dummies**, **The Complete Idiot's Guide to Growing Your Business With Google,** and **Wicked Cool Shell Scripts**, among others.

Although **Learning Unix for Mac OS X Tiger** is intended for Unix newbies, the content is surprisingly extensive and so well organized that this book can easily act as a refresher volume for even power Unix users, especially those new to Mac OS X.

In the first three chapters, Taylor quickly covers a great deal of ground as he introduces Unix, the CLI (command line interface) and Mac OS X Terminal basics, runs through an overview of the Mac filesystem (directory structure, pathnames) and delves into commands and optional switches related to navigating and exploring directories and files (`cd`, `ls`, `pwd`, etc.). The reader quickly becomes familiar with customizing the Terminal interface, entering commands and configuring advanced shell settings. Using the basic commands outlined in these chapters, Taylor also leads the reader through several important directories (`Library/Preferences` and `/var/log` for example) and takes some time explaining file permissions. He includes a good illustration of how to look at external volumes like attached iPods and digital cameras using `ls`.

Chapters 4 and 5 take the reader to file level management with explanations of file and directory name syntax; wildcard usage (`*`, `?`, `[]`, `{,}`); commands used to look inside files (`cat`, `less`, `grep`); command line editors (usage instructions for `vi` and brief overviews of `nano/pico` and `emacs`); how to organize files with `mkdir`, `cp`, `mv`, and `rm`; compressing and archiving files using `gzip` and `tar`; using `grep` for pattern matching within files; and finding files using `locate`, `find`, `mdls`, and `mdfind`.

Redirecting standard input and output is covered in Chapter 6. Taylor clarifies redirection, pipes and filters, the use of associated commands (`cat`, `tr`, `wc`, `head`, `tail`, `sort`, `uniq`), and piping output to a pager using `less`. He also discusses sending output to a printer via Unix commands (`lpstat` to look at configured printers, `lp` to add print jobs to a queue, `pr` to format files, `enscript` to translate plain text to Postscript, and `atprint` to send jobs via AppleTalk).

Chapter 7 explores monitoring multiple processes and discusses the concept of running commands as a background process using the `&` character, gathering running process information using `ps` and `top`, watching system processes through the system logs `using tail -f`, terminating processes using `kill` and `killall`, and interacting with GUI apps.

The next chapter shows the reader how to access remote systems via `ssh` and `rsh`, enabling web and ftp servers, and copying files between systems using `scp`, `rcp`, `curl`, and `ftp/sftp`. Taylor includes a brief aside on security and indicates other security issues where appropriate.

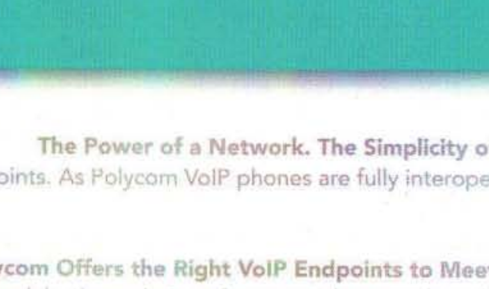
In Chapters 9 and 10, Taylor introduces X11, the X Window System, and how to find and install various open source applications using Fink, an open source software distribution and installation tool for Mac OS X. The author includes installing, using and customizing X11, and using remote access capabilities of X11. The author gives step-by-step instructions on how to install Fink, how to list and install packages and how to use the GUI front-end to Fink, FinkCommander. Taylor mentions but doesn't illustrate the Fink alternative, DarwinPorts perhaps because although command line junkies seem to prefer DarwinPorts, Fink's GUI interface makes it easier for beginners to maneuver and there are many more packages available through Fink. Other useful apps

THE FUTURE IS CALLING.

AND IT'S USING A POLYCOM VOIP PHONE.

The Power of a Network. The Simplicity of a Phone. Polycom® is a leading independent supplier of standards-based Voice over IP endpoints. As Polycom VoIP phones are fully interoperable with leading IP PBX and Softswitch platforms, you can have a variety of end-to-end, fully integrated VoIP solutions to choose from.

Polycom Offers the Right VoIP Endpoints to Meet Your Business Communication Needs. Polycom provides an assortment of VoIP phone models plus a choice of supported protocols and call management solutions to match the unique business communication requirements of your organization. The Polycom VoIP product portfolio includes the SoundPoint® IP family of desktop phones, an attendant console based on the SoundPoint IP 601 and Expansion Module, and the SoundStation® IP 4000 conference phone. Our phones are engineered to deliver a superb communications experience. They offer outstanding sound quality, advanced functionality, ease of use, simplified configuration and upgrades, and protection of your investment over time.



Please visit the Polycom Experience Center at www.polycom.com/experience to learn more about our VoIP products and other Polycom solutions to enhance your business communication needs.

POLYCOM®

VIDEO VOICE DATA WEB

TOGETHER, GREAT THINGS HAPPEN.

described: GIMP (GNU Image Manipulation Program), the email client Pine, the text-based Web browser Lynx, and GNU Backgammon. Taylor takes care to give the reader thorough details on downloading, installing and using most of the apps included in the chapter. Even a user completely unfamiliar with X11 or Fink can feel comfortable with the process outlined in these two sections.

In the final chapter, Taylor advises the reader on where to find additional resources through documentation (Help menus, `man` pages, web sites, and books), how to customize the Unix environment using aliases and functions, and programming options (shell scripts, AppleScript, Perl, Python, Ruby, C, and C++). This is a brief chapter intended to launch the reader into further study.

Learning Unix for Mac OS X Tiger follows O'Reilly's proven formula of providing superior books written by industry leaders. Font choices for both regular text and command line examples are highly readable with a large number of screenshots illustrating key points. Taylor's examples are representative of practical, familiar situations that even command line junkies will appreciate.

In addition to an excellent introduction to Unix for Tiger users, **Learning Unix for Mac OS X Tiger** is one of the best values around; readers can find this exceptional O'Reilly book for well under the listed $19.95 at a variety of book resellers.

Dave Taylor has a comfortable and relaxed style of writing that lends an air of ease to this complex subject. There is never a moment that the reader will feel lost. Each chapter is well thought out and executed and each topic follows the previous one in a reasonable fashion. The chapters are littered with useful examples of commands and optional switches and further illustrations of when they are most practically implemented.

Dave Taylor
280 pages
O'Reilly
ISBN: 0596009151
US $19.95

MT

About The Author

Mary Norbury is IT Director at the Barbara Davis Center for Childhood Diabetes, an affiliate center at the University of Colorado Health Sciences Center in Denver, Colorado. She has too-many-years-to-count experience in cross-platform systems implementation and administration in the education sector. You can reach her at norburym@mac.com.

Peace for your fingers

Speech Recognition for Mac OS X

iListen, the top-selling Speech Recognition system for Mac OS X, lets you do almost anything with your voice. Our natural speech engine and voice command & control system takes the pain out of typing.

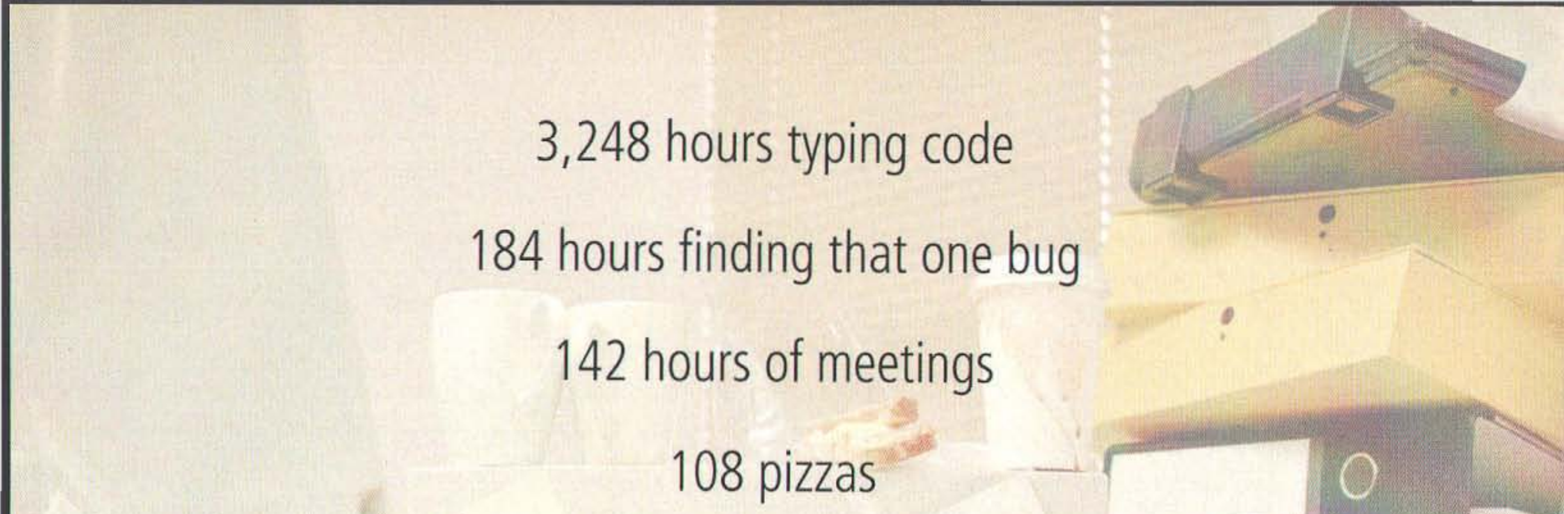
Optional unique ScriptPaks let you select all your menus by voice, there's even a ScriptPak for BBEdit, the developers choice of text and source editing.

For a limited time you can get a free BBEdit ScriptPak with any purchase of iListen from our online store. The BBEdit ScriptPak allows you to do almost anything in BBEdit with your voice that you would normally do with your keyboard and mouse. To get the free ScriptPak you must use the coupon code FREE-BBEDIT when checking out.

See www.macspeech.com/promo/bbedit.html or call (603) 251-1477 for more details.

MacSpeech

www.macspeech.com



The new HASP family of products is the next generation in protection ensuring the highest level of security for your software. It provides an easy set of tools to automatically protect your software and implement new and innovative licensing options.

▸ Strongest anti-piracy solution:

Our powerful, 128-bit AES encryption provides a strong locking mechanism that ensures you get paid for every copy of your software.

ENCRYPTED DATA
PROTECTED SOFTWARE
AES ENCRYPTION ENGINE
DECRYPTED DATA

▸ Robust envelope:

Automatically protects Windows executables, DLLs, .Net and Mac applications, securing the weakest link and ensuring your valuable IP remains unreachable. Multiple protection layers provide unequaled security.

▸ Best overall solution as rated by an independent testing lab:

In KeyLabs tests, HASP HL outperformed the competition in security, ease-of-use, flexibility of tools and functionality, and compatibility across platforms and environments. A copy of the full KeyLabs report is available from our Web site at www.aladdin.com/hasp.

Category	Weight	Aladdin HASP HL	Sentinel UltraPro
Security	35%	85.1%	60.9%
Ease-of-use	25%	93.7%	77%
Licensing	20%	75%	75%
Flexibility of tools and functionality	10%	93.3%	71.2%
Compatibility across platforms and environments	10%	93.3%	71%
Overall Evaluation		87.3%	69.8%

Request a Software Developer Kit today at www.aladdin.com/hasp

Aladdin®
SECURING THE GLOBAL VILLAGE

North America: 1-800-562-2543, 847-818-3800 • UK • Germany • Israel • Benelux • France • Spain • Asia Pacific • Japan

Advertiser/Product Index

The index on this page is provided as a service to our readers. The publisher does not assume any liability for errors or omissions.

How to Stop Racing the Clock.

☐Work Longer? ☐Work Harder?

☑ Energize Your Mac!

We know your day keeps getting longer and longer. With every release of software, your Mac is bogged down even more. With every click, there's a pause. You find yourself working longer, working harder. *A faster Mac means that you can work faster, not harder - be more productive!*

Let the original Mac Performance Shop help. Daystar has been creating Mac speed for over 16 years. Whether your bottleneck is storage, connectivity or just raw CPU speed, we deliver the performance you need, where you need it.

CPU Upgrades for Raw Speed. We upgrade any Power Macintosh, any iMac Flat Panel, any PowerBook G3 and some PowerBook G4s.

Fast and Large Storage for Real-Time Video. Our **TURBO***SATA* solutions can make your drives perform like RAM. Projects open in a flash and edit in real-time.

Extreme Wireless. Wireless is great, unless you're getting slow transfers. Even Airport Extreme's are slow when the signal is weak. Daystar can boost your signals and energize your wireless network.

But, if You Really need a G5? Daystar is the only Mac Performance Manufacturer that is also an Apple Authorized Reseller. Not only can you trade-in your system for the latest and greatest... but the Daystar Pro's can upgrade it for maximum performance!

Call 877-439-8646 and beat the clock.

Authorized Reseller

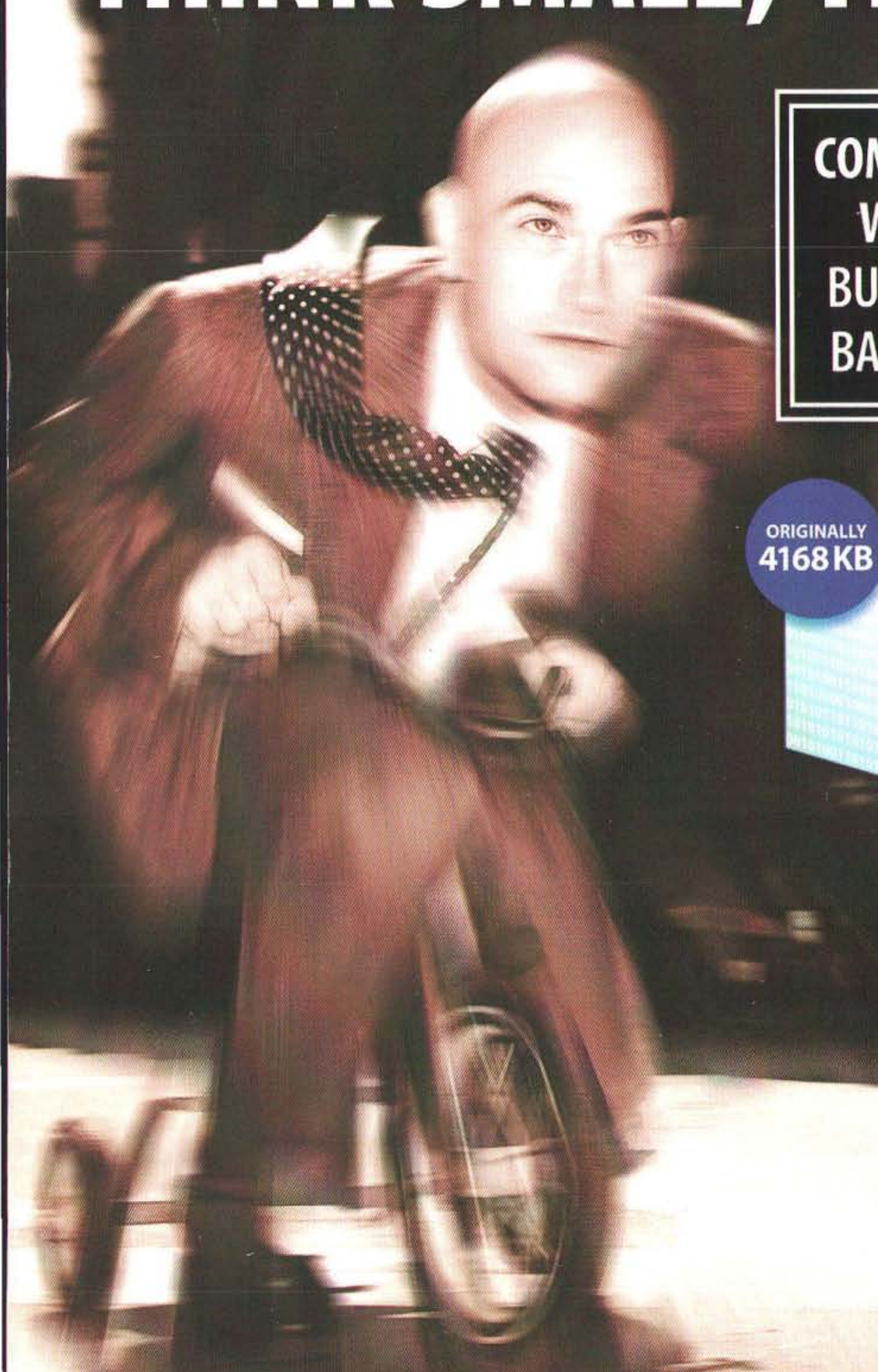
Daystar TECHNOLOGY

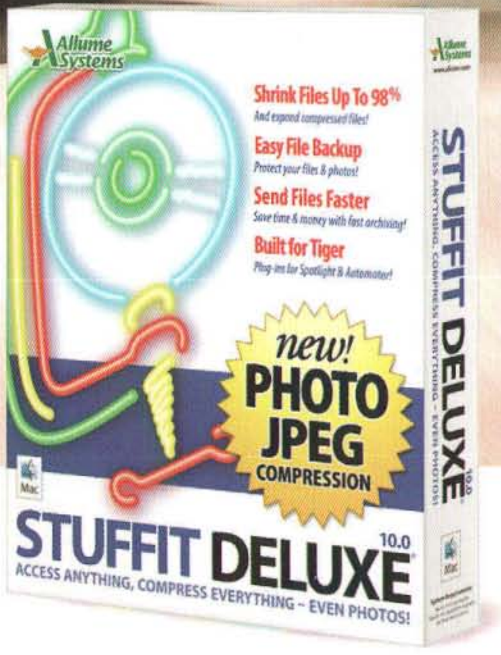
Daystar Technology - Your Macintosh Performance Shop
5018 Bristol Industrial Way, #202, Buford, GA 30518 USA
Toll Free: 877-439-8646 or 770-614-5400

Daystar-Tech.com **Daystar-Forum.com** **Daystar-Store.com**

THINK SMALL, THINK *FAST.*

COMPRESS JPEGS UP TO 30%
WITH NO QUALITY LOSS!
BUILT FOR MAC OS X "TIGER"
BACKUP TO .MAC, CD, & DVD




Allume Systems
Shrink Files Up To 98%
And expand compressed files!
Easy File Backup
Protect your files & photos!
Send Files Faster
Save time & money with fast archiving!
Built for Tiger
Plug-ins for Spotlight & Automator!
new!
PHOTO
JPEG
COMPRESSION
STUFFIT DELUXE 10.0
ACCESS ANYTHING, COMPRESS EVERYTHING – EVEN PHOTOS!

Allume Systems
A Division of Smith Micro Software

Available from your favorite retailers and catalogs. Download StuffIt Expander for FREE at www.stuffit.com!

Download a FREE trial version of StuffIt Standard Edition at **www.allume.com.** For information about StuffIt's industry-leading JPEG compression, please visit **www.stuffit.com.**